

ΙΟΝΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ



ΒΑΣΙΛΕΙΑΔΗΣ ΘΩΜΑΣ

Π2013091

ΑΝΑΦΟΡΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ
«ΠΡΟΣΟΜΟΙΩΣΗ ΚΑΙ ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΜΙΑΣ ΕΠΙΔΗΜΙΑΣ»

Επιβλέπουσα Επιτροπή:

Βλάμος Παναγιώτης (Επιβλέπων Καθηγητής)

Κουροθανάσης Παναγιώτης

Μάγκος Εμμανουήλ

Κέρκυρα, Οκτώβριος 2019

ΑΝΑΦΟΡΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ
«ΠΡΟΣΟΜΟΙΩΣΗ ΚΑΙ ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΜΙΑΣ ΕΠΙΔΗΜΙΑΣ»

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΕΡΙΛΗΨΗ.....	2
ΕΙΣΑΓΩΓΗ	3
ΚΕΦΑΛΑΙΟ 1: ΠΡΟΣΟΜΟΙΩΣΗ ΚΑΙ ΜΟΝΤΕΛΟΠΟΙΗΣΗ.....	6
1.1 Η ΕΝΝΟΙΑ ΤΗΣ ΠΡΟΣΟΜΟΙΩΣΗΣ ΚΑΙ ΤΗΣ ΜΟΝΤΕΛΟΠΟΙΗΣΗΣ.....	6
1.2 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ	9
1.3 ΣΥΣΤΗΜΑ	15
1.3.1 ΚΑΤΗΓΟΡΙΕΣ ΣΥΣΤΗΜΑΤΩΝ	15
1.4 ΜΟΝΤΕΛΟ.....	17
1.4.1 ΚΑΤΗΓΟΡΙΕΣ ΜΟΝΤΕΛΩΝ	19
1.4.2 ΤΕΧΝΙΚΕΣ ΕΥΡΕΣΗΣ ΜΟΝΤΕΛΟΥ	19
1.4.3 ΚΑΤΑΣΚΕΥΗ ΜΟΝΤΕΛΩΝ ΠΡΟΣΟΜΟΙΩΣΗΣ	20
1.5 ΕΦΑΡΜΟΓΕΣ ΤΗΣ ΠΡΟΣΟΜΟΙΩΣΗΣ	21
ΚΕΦΑΛΑΙΟ 2: ΜΟΝΤΕΛΑ ΕΠΙΔΗΜΙΟΛΟΓΙΑΣ.....	25
2.1 Η ΕΝΝΟΙΑ ΤΗΣ ΕΠΙΔΗΜΙΟΛΟΓΙΑΣ	25
2.2 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ	27
2.3 ΤΟ ΜΟΝΤΕΛΟ SIR.....	29
2.4 ΤΟ ΜΟΝΤΕΛΟ SIR ΧΩΡΙΣ ΖΩΤΙΚΗ ΔΥΝΑΜΙΚΗ	30
2.5 ΤΟ ΜΟΝΤΕΛΟ SIR ΜΕ ΖΩΤΙΚΗ ΔΥΝΑΜΙΚΗ	33
2.6 ΤΟ ΜΟΝΤΕΛΟ SIS	34
2.7 ΤΟ ΜΟΝΤΕΛΟ SEIR	35
2.8 ΤΟ ΜΟΝΤΕΛΟ SEIS.....	37
2.9 ΤΟ ΜΟΝΤΕΛΟ MSEIR.....	38
2.10 ΚΑΤΩΤΑΤΑ ΟΡΙΑ ΓΙΑ ΑΥΘΑΙΡΕΤΑ ΜΟΝΤΕΛΑ ΚΑΤΑΡΡΑΚΤΗ ΣΕ ΑΥΘΑΙΡΕΤΑ ΔΙΚΤΥΑ.....	39
2.11 ΜΟΝΤΕΛΟ ΣΥΝΕΧΟΥΣ ΧΡΟΝΟΥ WINNER TAKES ALL.....	43
ΚΕΦΑΛΑΙΟ 3: ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΡΥΘΜΟΝ	48
3.1 ΕΙΣΑΓΩΓΗ	48
3.2 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ	49
3.3 ΔΙΕΡΜΗΝΕΥΤΗΣ.....	50
3.4 IDLE	50
3.5 ΔΟΜΗ ΚΑΙ ΑΝΑΠΤΥΞΗ	52

3.5.1 ΟΝΟΜΑΤΟΔΟΣΙΑ ΜΕΤΑΒΛΗΤΩΝ	52
3.5.2 ΤΥΠΟΙ ΔΕΔΟΜΕΝΩΝ	53
3.5.3 ΑΡΧΕΓΟΝΟΙ ΤΥΠΟΙ ΔΕΔΟΜΕΝΩΝ	53
3.5.4 ΛΙΣΤΑ.....	54
3.5.5 ΠΛΕΙΑΔΑ.....	54
3.5.6 ΛΕΞΙΚΟ	55
3.5.7 ΑΝΑΘΕΣΗ ΤΙΜΩΝ	55
3.5.8 ΕΣΟΧΕΣ	56
3.5.9 ΤΕΛΕΣΤΕΣ.....	57
3.5.10 ΕΜΒΕΛΕΙΑ ΟΝΟΜΑΤΩΝ ΟΝΤΟΤΗΤΩΝ.....	62
3.5.11 ΣΧΟΛΙΑ	63
3.5.12 ΔΗΛΩΣΗ GLOBAL.....	63
3.6 ΠΑΡΑΔΕΙΓΜΑ ΠΡΟΓΡΑΜΜΑΤΟΣ	64
3.6.1 ΔΙΑΔΙΚΑΣΤΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ.....	64
3.6.2 ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΗΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ	65
ΚΕΦΑΛΑΙΟ 4: ΠΡΟΓΡΑΜΜΑ ΠΡΟΣΟΜΟΙΩΣΗΣ ΚΑΙ ΜΟΝΤΕΛΟΠΟΙΗΣΗΣ	
ΜΙΑΣ ΕΠΙΔΗΜΙΑΣ.....	66
4.1 ΤΟ ΠΡΟΓΡΑΜΜΑ.....	66
4.2 ΠΕΡΙΓΡΑΦΗ ΚΩΔΙΚΑ	74
4.3 ΣΥΜΠΕΡΑΣΜΑΤΑ	79
ΕΛΛΗΝΙΚΗ ΒΙΒΛΙΟΓΡΑΦΙΑ	81
ΞΕΝΟΓΛΩΣΣΗ ΒΙΒΛΙΟΓΡΑΦΙΑ	82
ΠΑΡΑΡΤΗΜΑ ΕΙΚΟΝΩΝ.....	84

ΠΡΟΛΟΓΟΣ - ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα πτυχιακή εργασία λαμβάνει χώρα στα πλαίσια σπουδών του Ιονίου Πανεπιστημίου, τμήματος Πληροφορικής, με θέμα «Προσομοίωση και μοντελοποίηση μιας επιδημίας» κάτω από την τριμελή επιτήρηση των καθηγητών Βλάμο Παναγιώτη (επιβλέπων καθηγητής), Κουροθανάση Παναγιώτη, Μάγκο Εμμανουήλ. Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή Βλάμο Παναγιώτη για την πολύτιμη βοήθειά του και καθοδήγηση κατά την διάρκεια εκπόνησης της παρούσας πτυχιακής εργασίας. Επίσης θα ήθελα να ευχαριστήσω τους καθηγητές Κουροθανάση Παναγιώτη και Μάγκο Εμμανουήλ, για την θερμή συνεργασία και αποδοχή στο να είναι μέλη της επιβλέπουσας επιτροπής της πτυχιακής μου.

ΠΕΡΙΛΗΨΗ

Η παρούσα εργασία, πρόκειται για την αναφορά της πτυχιακής εργασίας που υλοποιήθηκε με θέμα «Προσομοίωση και μοντελοποίηση μιας επιδημίας». Σκοπός της είναι η μελέτη και ανάλυση των εννοιών προσομοίωση, μοντελοποίηση, καθώς και να επεξηγηθεί η διαφορετικότητα αυτών των δύο εννοιών, αλλά και την διαφορά της έννοιας προσομοίωση με την έννοια εξομοίωση. Στη συνέχεια, μετά από εκτενή ανάλυση των εννοιών σύστημα και μοντέλο, καθώς και πώς κατασκευάζεται ένα μοντέλο προσομοίωσης, γίνεται μελέτη της «προσομοίωσης και μοντελοποίησης» σαν μια έννοια. Έπειτα, ακολουθεί εισαγωγή στην επιδημιολογία και τις επιδημίες, καθώς και στα επιδημιολογικά μοντέλα SIR, SIS, SEIS, SEIR και MSEIR. Ακόμη, παρουσιάζονται και κάποια άλλα χρήσιμα, συνεχούς αλλά και διακριτού χρόνου, επιδημιολογικά μοντέλα και γίνονται παραδείγματα πάνω σε αυτά. Συνεχίζοντας, γίνεται εισαγωγή στην γλώσσα προγραμματισμού Python πάνω στην οποία υλοποιήθηκε η πτυχιακή εργασία (πρόγραμμα), και αναλύονται τα χαρακτηριστικά της, αλλά και η λειτουργικότητα της. Παράλληλα, δίνονται παραδείγματα προγραμμάτων για την καλύτερη κατανόηση των χαρακτηριστικών της Python. Έπειτα, γίνεται ανάλυση της μεθοδολογίας που ακολουθήθηκε και της λειτουργικότητας του προγράμματος που υλοποιήθηκε στα πλαίσια της πτυχιακής εργασίας, και εν τέλει διεξάγονται τα συμπεράσματα.

ΕΙΣΑΓΩΓΗ

Η παρούσα έκθεση αποτελεί την γραπτή αναφορά της πτυχιακής άσκησης, η οποία διεξήχθη στο πλαίσιο του προγράμματος σπουδών του Ιονίου Πανεπιστημίου με τίτλο «Προσομοίωση και Μοντελοποίηση μιας επιδημίας».

Με το πέρασμα των χρόνων, καθώς και την ραγδαία ανάπτυξη της τεχνολογίας, από εκεί που κάποτε ο άνθρωπος έπρεπε να δοκιμάσει αυτοπροσώπως κάτι για να πάρει αποτελέσματα και να τα αναλύσει για να βγάλει συμπεράσματα, πλέον όλα αυτά μπορούν να γίνουν με την βοήθεια υπολογιστικών μηχανών και ενός μεγάλου όγκου μαθηματικών μοντέλων που εφαρμόζονται σε μια «διαδικασία» που την αποκαλούμε «προσομοίωση και μοντελοποίηση».

Αντικείμενο της παρούσας εργασίας είναι η προσομοίωση και μοντελοποίηση μιας επιδημίας σε ένα δοκιμαστικό πληθυσμό, και ο τρόπος με τον οποίο υλοποιήθηκε είναι με την εφαρμογή επιδημιολογικών μαθηματικών μοντέλων πάνω στον προγραμματισμό, κατασκευάζοντας ένα πρόγραμμα με την χρήση Python, το οποίο μοντελοποιεί έναν δοκιμαστικό πληθυσμό, και συλλέγει όλα τα απαραίτητα στοιχεία που απαιτούνται, και στην συνέχεια προσομοιώνει την εξάπλωση μιας επιδημίας πάνω σε αυτό τον δοκιμαστικό πληθυσμό, με σκοπό την διεξαγωγή συμπερασμάτων, τα οποία μπορούν να χρησιμοποιηθούν για αναλυθεί ο τρόπος με τον οποίο θα μπορούσε να εξαπλωθεί μια επιδημία σε ένα πραγματικό πληθυσμό π.χ της πόλης της Κέρκυρας.

Στόχοι της παρούσας εργασίας είναι να αναλυθούν εις βάθος οι έννοιες της προσομοίωσης, της μοντελοποίησης, καθώς και την διαφορά μεταξύ της προσομοίωσης με την εξομοίωση. Σταδιακά, αναλύοντας όλες τις έννοιες ξεχωριστά, καταλήγουμε να σχηματίζουμε την έννοια της «προσομοίωσης και μοντελοποίησης» συνολικά. Είναι πολύ σημαντικό να αναλυθεί κάθε έννοια ξεχωριστά καθώς είναι όλες εντελώς διαφορετικές έννοιες και πολλές φορές η σημασίες τους συγχέονται. Παράλληλα, αναλύθηκαν τα μοντέλα προσομοίωσης καθώς και πως κατασκευάζεται ένα τέτοιο μοντέλο, και μελετήθηκε η επιδημιολογία σαν επιστήμη αλλά και σαν έννοια. Ακόμη, αναλύονται διεξοδικά οι έννοιες του μοντέλου και του συστήματος. Επίσης, στην παρούσα πτυχιακή εργασία αναλύονται αρκετά μαθηματικά επιδημιολογικά, ένα από τα

οποία είναι το μοντέλο SIR, το οποίο ένα από τα πιο βασικά επιδημιολογικά μοντέλα για τη μελέτη ενός πληθυσμού υπό την έξαρση μίας ασθένειας και προσφέρει συμπεράσματα χρήσιμα για την πρόβλεψη της εξέλιξης της.

Ένα άλλο επιδημιολογικό μοντέλο που αναλύεται είναι το μοντέλο SIS, το οποίο μοιάζει αρκετά με το μοντέλο SIR, αλλά διαφέρει ελαφρώς διότι είναι μοντέλο που υποδηλώνει ασθένειες στις οποίες δεν έχουν χορηγηθεί ανοσοποιητικά στην ασθένεια τα ανακτημένα άτομα και είναι και πάλι ευαίσθητα στη μόλυνση μετά την αποκατάσταση. Το μοντέλο SIS μπορεί να χρησιμοποιηθεί για να περιγράψει σεξουαλικά μεταδιδόμενα νοσήματα όπως η σύφιλη ή η γονόρροια στα οποία τα άτομα αντιμετωπίζονται και ανακτώνται αλλά είναι αμέσως ευαίσθητα και πάλι. Συνεχίζοντας ένα από τα μοντέλα που αναλύονται είναι και το SEIR, το οποίο μοντελοποιεί τις ροές ανθρώπων μεταξύ τεσσάρων καταστάσεων: επιδεκτικές (S), εκτεθειμένες (E), μολυσμένες (I) και ανθεκτικές (R). Κάθε μία από αυτές τις μεταβλητές αντιπροσωπεύει τον αριθμό των ατόμων που ανήκουν σε αυτές τις ομάδες. Το SEIR διαφέρει από το μοντέλο SIR στην προσθήκη μιας περιόδου λανθάνουσας περιόδου. Τα άτομα που είναι εκτεθειμένα (E) έχουν έρθει σε επαφή με ένα μολυσμένο άτομο, αλλά δεν είναι μολυσματικά. Επίσης αναλύεται το μοντέλο SEIS. Το ευαίσθητο (S), εκτεθειμένο (E), μολυσματικό (I), ευαίσθητο (S) μοντέλο (SEIS) είναι ευρέως γνωστό στη μαθηματική επιδημιολογία ως μοντέλο λοίμωξης στο οποίο υπάρχει μια λανθάνουσα περίοδος μεταξύ της στιγμής της μόλυνσης και της εμφάνισης της μολυσματικότητας. Το μοντέλο διαμερίσματος έχει μελετηθεί καλά, αλλά το αντίστοιχο σύστημα σωματιδίων δεν έχει λάβει μέχρι τώρα καμία προσοχή. Για το μοντέλο του συστήματος σωματιδίων σε μία χωρική διάσταση, δίνεται ανώτερα και κατώτερα όρια στις κρίσιμες τιμές, αποδεικνύεται η σύγκλιση κρίσιμων τιμών στο όριο του μικρού και μεγάλου λανθάνοντος χρόνου και προσδιορίζεται μια περιοριστική διαδικασία στην οποία το μοντέλο SEIS συγκλίνει στο όριο μεγάλου λανθάνοντος χρόνου. Το μοντέλο SIR είναι ένα βασικό μοντέλο επιδημίας που κατατάσσει έναν πληθυσμό σε τρεις υποομάδες: ευαίσθητο S, μολυσμένο I και απομακρυσμένο R. Αυτό το μοντέλο όμως δεν λαμβάνει υπόψη τη χωρική κατανομή κάθε υποομάδας αλλά θεωρεί ότι ο συνολικός αριθμός ατόμων που ανήκουν σε κάθε υποομάδα. Υπάρχουν πολλές παραλλαγές του μοντέλου SIR. Για παράδειγμα, για ένα έκθετο (E) διαμέρισμα, όταν υπάρχει σημαντική περίοδος επώασης κατά τη διάρκεια της

οποίας το άτομο έχει μολυνθεί αλλά δεν έχει ακόμη συμπτώματα, και μια μητρική προερχόμενη ανοσία (M), εξετάζει νεογνά που είναι άνοσα κατά τους πρώτους μήνες ζωής ως αποτέλεσμα μητρικών αντισωμάτων. Το μοντέλο που λαμβάνει υπόψη όλα αυτά τα διαμερίσματα αναφέρεται ως μοντέλο MSEIR. Επίσης, για τη μελέτη της χωρικής κατανομής, συχνά εισήχθησαν στοχαστικές διαδικασίες για να περιγράψουν τη διασπορά των ατόμων. Τέτοιες υποθέσεις δεν φαίνονται να ισχύουν για τον άνθρωπο, επειδή σχεδόν όλοι κινούνται μέσα σε μια μικρή σταθερή ακτίνα στην πράξη. Ακόμη και αν τα άτομα δεν διασκορπίζονται, η μετάδοση της ασθένειας συμβαίνει. Έτσι, προτείνουμε απλά συνεχή και διακριτά μοντέλα SIR που παρουσιάζουν χωρικές κατανομές. Τα αποτελέσματα των προσομοιώσεων αποδεικνύουν ότι η ταχύτητα διάδοσης και το μέγεθος της επιδημίας εξαρτώνται από την πυκνότητα του πληθυσμού και τη μολυσματική ακτίνα. Για όλα αυτά τα μοντέλα τα οποία αναλύονται, δίνονται παραδείγματα και εφαρμογές αυτών.

ΚΕΦΑΛΑΙΟ 1: ΠΡΟΣΟΜΟΙΩΣΗ ΚΑΙ ΜΟΝΤΕΛΟΠΟΙΗΣΗ

Σε αυτό το κεφάλαιο θα γίνει μια ανάλυση των εννοιών προσομοίωση και μοντελοποίηση. Επίσης θα γίνει επεξήγηση της διαφοράς των εννοιών προσομοίωση και εξομοίωση.

Η προσομοίωση είναι μια τεχνική μοντελοποίησης μέσα από την οποία η επιστήμη της Πληροφορικής με την επιστήμη της Στατιστικής συνδέονται και συνεργάζονται για την επίτευξη της προσομοίωσης. Η μεθοδολογία έχει ως βασικό χαρακτηριστικό την αποτύπωση ενός συστήματος με τη βοήθεια λογικών διαγραμμάτων, σχέσεων και προγραμμάτων στον υπολογιστή και εν τέλει της λήψης βέλτιστων αποφάσεων με την εκτέλεση πειραμάτων, τη δειγματοληψία και την ανάλυση δεδομένων.

1.1 Η ΕΝΝΟΙΑ ΤΗΣ ΠΡΟΣΟΜΟΙΩΣΗΣ ΚΑΙ ΤΗΣ ΜΟΝΤΕΛΟΠΟΙΗΣΗΣ

Σε αυτή την υποενότητα θα γίνει ανάλυση της μοντελοποίησης, της προσομοίωσης καθώς και θα εξηγηθεί η διαφορά μεταξύ προσομοίωσης και εξομοίωσης. Συμπερασματικά, από τον ορισμό του μοντέλου μπορεί να ειπωθεί ο ορισμός μοντελοποίηση, που είναι η διαδικασία εύρεσης του μοντέλου.

Ο Μ. Ρουμελιώτης (2001) ανέφερε ότι η δημιουργία μοντέλων για προσομοίωση έχει ως στόχους:

- Τη μελέτη και τη συμπεριφορά ενός συστήματος
- Τον έλεγχο υποθέσεων ή θεωριών για την παρατηρούμενη συμπεριφορά ενός συστήματος
- Την πρόβλεψη ή εκτίμηση της μελλοντικής συμπεριφοράς ενός συστήματος.

Ο Mc Graw-Hill (2003) έδωσε τον ορισμό της προσομοίωσης ως τη μίμηση της συμπεριφοράς ενός συστήματος συνήθως με τη χρήση ηλεκτρονικού υπολογιστή ή με τη χρήση ειδικού εξοπλισμού.

Ο ηλεκτρονικός υπολογιστής είναι το καλύτερο εργαλείο προσομοίωσης διότι υποστηρίζει πολλά προγράμματα που διαθέτουν πολλές δυνατότητες για προσομοίωση

όπως είναι το matlab, mathematica, maple και υπάρχουν και άλλα πολλά, αλλά αυτά είναι τα πιο διαδεδομένα.

Όπως ανέφερε ο Μ. Ρουμελιώτης (2001), η προσομοίωση αποτελείται από **3 φάσεις**:

1. Η κατασκευή του μοντέλου προσομοίωσης στον ηλεκτρονικό υπολογιστή.
2. Η εκτέλεση ή το τρέξιμο του μοντέλου στον ηλεκτρονικό υπολογιστή.
3. Η ανάλυση των αποτελεσμάτων της προσομοίωσης.

Με τη χρήση της **προσομοίωσης**¹ προκύπτουν κάποια θετικά χαρακτηριστικά, που είναι:

- Η προσομοίωση είναι οικονομική, ασφαλής και ηθική διότι μπορούν να διεξαχθούν επικίνδυνα πειράματα συστημάτων στον εικονικό κόσμο αντί στον πραγματικό.
- Η προσομοίωση πλέον ανταποκρίνεται όλο και πιο πολύ στη πραγματικότητα διότι επιτρέπει τη διαμόρφωση παραμέτρων του περιβάλλοντος στο οποίο βρίσκεται το σύστημα που αναπαρίσταται από το μοντέλο.
- Η προσομοίωση διεξάγεται πιο γρήγορα από τον πραγματικό χρόνο, που επιτρέπει αποτελεσματικότερη ανάλυση και δημιουργία εναλλακτικών λύσεων για τη βελτιστοποίηση του συστήματος που αναπαριστάτε από το μοντέλο.
- Η προσομοίωση επιτρέπει την ανάλυση και ενσωμάτωση συστημάτων που ο συνδυασμός αυτών αποτελούν ένα τελικό σύστημα.

Από την άλλη, εξομοίωση είναι μια ιδεατή ή εικονική μηχανή που μπορεί να περιγράψει την συμπεριφορά ή λειτουργία μιας πραγματικής μηχανής ή διάταξης ή συστήματος μηχανής-περιβάλλοντος. Για να επισημανθεί η διαφορά της προσομοίωσης και εξομοίωσης θα δοθεί ένα απλό παράδειγμα προσομοίωσης και εξομοίωσης:

Οι πιλότοι, κατά την διάρκεια της εκπαίδευσής τους, καλούνται να χρησιμοποιήσουν ένα μηχάνημα, το οποίο δίνει την αίσθηση στον πιλότο ότι πετάει. Αυτό το μηχάνημα

¹ Modeling and Simulation https://en.wikipedia.org/wiki/Modeling_and_simulation , αναρτήθηκε 24 Απριλίου 2009, τελευταία πρόσβαση 15/9/2019.

ονομάζεται **εξομοιωτής πτήσεων**. Ενώ το λογισμικό που παρουσιάζει ένα βίντεο πτήσης και δεν δίνει καμία αίσθηση ότι ο πιλότος πιλοτάρει το αεροπλάνο, ονομάζεται **προσομοιωτής πτήσεων**.

Αφού δόθηκε η εξήγηση της έννοιας μοντελοποίησης όπως και της προσομοίωσης και διαφοράς μεταξύ προσομοίωσης και εξομοίωσης, πλέον μπορεί να δοθεί ολοκληρωμένος ο ορισμός της μοντελοποίησης και προσομοίωσης.

Η **μοντελοποίηση και η προσομοίωση**² είναι η χρήση μοντέλων (π.χ. φυσικής, μαθηματικής ή λογικής αναπαράστασης ενός συστήματος, οντότητας, φαινομένου ή διεργασίας) ως βάση για προσομοιώσεις για την ανάπτυξη δεδομένων που χρησιμοποιούνται για διοικητική ή τεχνική λήψη αποφάσεων. Στην εφαρμογή ηλεκτρονικού υπολογιστή "Μοντελοποίηση και προσομοίωση", ένας υπολογιστής χρησιμοποιείται για την κατασκευή ενός μαθηματικού μοντέλου που περιέχει βασικές παραμέτρους του φυσικού μοντέλου. Το μαθηματικό μοντέλο αντιπροσωπεύει το φυσικό μοντέλο σε εικονική μορφή και εφαρμόζονται οι συνθήκες που ορίζουν το πείραμα ενδιαφέροντος. Ξεκινάει η προσομοίωση, δηλαδή ο υπολογιστής υπολογίζει τα αποτελέσματα αυτών των συνθηκών στο μαθηματικό μοντέλο, και τα αποτελέσματα εξάγονται σε μορφή που είναι είτε αναγνώσιμη από το μηχάνημα είτε από άνθρωπο, ανάλογα με την υλοποίηση. Η χρήση της προσομοίωσης και της μοντελοποίησης στο χώρο της μηχανικής είναι γνωστή. Η τεχνολογία προσομοίωσης ανήκει στο σύνολο εργαλείων των μηχανικών όλων των τομέων εφαρμογής και έχει συμπεριληφθεί στο σώμα γνώσεων της διαχείρισης μηχανικών.

Η προσομοίωση και μοντελοποίηση συμβάλλει στα εξής:

- Στη μείωση του κόστους.
- Στην αύξηση της ποιότητας των προϊόντων και συστημάτων.
- Στην τεκμηρίωση και την αρχειοθέτηση των διδαγμάτων.

Επειδή τα αποτελέσματα μιας προσομοίωσης είναι τόσο καλά όσο το υποκείμενο μοντέλο, οι μηχανικοί, οι χειριστές και οι αναλυτές πρέπει να δώσουν ιδιαίτερη προσοχή

² Modeling and Simulation https://en.wikipedia.org/wiki/Modeling_and_simulation , αναρτήθηκε 24 Απριλίου 2009, τελευταία πρόσβαση 19/5/2019.

στην κατασκευή του. Για να διασφαλιστεί ότι τα αποτελέσματα της προσομοίωσης είναι εφαρμόσιμα στον πραγματικό κόσμο, ο χρήστης πρέπει να κατανοήσει τις υποθέσεις, τις ιδέες και τους περιορισμούς της εφαρμογής του. Επιπλέον, τα μοντέλα μπορούν να ενημερωθούν και να βελτιωθούν χρησιμοποιώντας αποτελέσματα πραγματικών πειραμάτων.

Η χρήση τέτοιων μαθηματικών μοντέλων και προσομοιώσεων αποφεύγει τον πραγματικό πειραματισμό, ο οποίος μπορεί να είναι δαπανηρός και χρονοβόρος. Αντί αυτού, οι μαθηματικές γνώσεις και η υπολογιστική ισχύς χρησιμοποιούνται για την επίλυση πραγματικών προβλημάτων με φθινό και χρονικά αποδοτικό τρόπο. Ως εκ τούτου, η προσομοίωση και μοντελοποίηση μπορεί να διευκολύνει την κατανόηση της συμπεριφοράς του συστήματος χωρίς να δοκιμάζει πραγματικά το σύστημα στον πραγματικό κόσμο. Για παράδειγμα, για να προσδιοριστεί ποιος τύπος σπόιλερ θα βελτιώνει την πρόσφυση κατά το σχεδιασμό ενός αγωνιστικού αυτοκινήτου, θα μπορούσε να χρησιμοποιηθεί μια προσομοίωση του αυτοκινήτου για την εκτίμηση της επίδρασης διαφορετικών σχημάτων σπορ στο συντελεστή τριβής σε μια στροφή. Χρήσιμες ιδέες για τις διαφορετικές αποφάσεις στο σχεδιασμό θα μπορούσαν να συλλεχθούν χωρίς πραγματικά να κατασκευάσουν το αυτοκίνητο. Επιπλέον, η προσομοίωση μπορεί να υποστηρίξει πειραματισμούς που εμφανίζονται συνολικά σε λογισμικό ή σε περιβάλλοντα ανθρώπου-σε-βρόχο όπου η προσομοίωση αντιπροσωπεύει συστήματα ή παράγει δεδομένα που απαιτούνται για την επίτευξη των στόχων του πειράματος. Επιπλέον, η προσομοίωση μπορεί να χρησιμοποιηθεί για την εκπαίδευση ατόμων που χρησιμοποιούν ένα εικονικό περιβάλλον το οποίο διαφορετικά θα ήταν δύσκολο ή ακριβό να παραχθεί.

1.2 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ

Με το πέρασμα των χρόνων, η προσομοίωση χρησιμοποιείται ευρέως ως εργαλείο για την αύξηση της παραγωγικότητας.

Το 1999, το λογισμικό προσομοίωσης που χρησιμοποιήθηκε από την Cymer Inc. αύξησε την ικανότητα παραγωγής της εταιρείας από 5 μονάδες τον μήνα στις αρχές του

1999 45 μονάδες τον μήνα στα τέλη του 1999, σημειώνοντας αύξηση κατά περίπου 400%. Τα γραφικά περιβάλλοντα έχουν αναμφίβολα επιφέρει μεγάλες επιπτώσεις σε εταιρίες προσομοίωσης. Μοντέλα εύκολα στην χρήση έχουν ως αποτέλεσμα σε πακέτα χαμηλού κόστους, κάτι το οποίο πριν μερικά χρόνια ήταν αδιανόητο. Η ιστορία της προσομοίωσης του υπολογιστή χρονολογείται από τον Β Παγκόσμιο Πόλεμο όταν δύο μαθηματικοί Jon Von Neumann και Stanislaw Ulam αντιμετώπισαν το πρόβλημα της συμπεριφοράς των νετρονίων. Οι πειραματισμοί και οι δοκιμές ήταν πολύ δαπανηροί και το πρόβλημα ήταν πολύ περίπλοκο για να αναλυθεί. Ως εκ τούτου, προτάθηκε από του μαθηματικούς της εποχής, η τεχνική τροχιών ρουλέτας. Τα βασικά δεδομένα που αφορούσαν την εμφάνιση διαφόρων γεγονότων ήταν γνωστά, στα οποία οι πιθανότητες χωριστών γεγονότων συγχωνεύθηκαν σε μία βήμα-προς-βήμα ανάλυση, η οποία είχε σκοπό να προβλέψει το αποτέλεσμα όλης της σειράς γεγονότων. Μετά την μεγάλη επιτυχία των τεχνικών πάνω στο πρόβλημα των νετρονίων, η προσομοίωση σύντομα έγινε δημοφιλής και είχε πολλές εφαρμογές σε διάφορους τομείς.

Στα τέλη του 1940 και στις αρχές του 1950, οι εμπορικά σχεδιασμένοι ηλεκτρονικοί υπολογιστές άρχισαν να εμφανίζονται τόσο σε αναλογικές όσο και σε ψηφιακές συσκευές σε ορισμένους οργανισμούς. Τα μέλη του τεχνικού προσωπικού αυτών των οργανισμών βρέθηκαν ξαφνικά υπεύθυνα να αποφασίσουν πώς να χρησιμοποιήσουν αυτά τα ηλεκτρονικά τέρατα και να τα χρησιμοποιήσουν για προβλήματα της καθημερινότητας. Ένας μηχανικός που εργαζόταν στο Κέντρο Δοκιμών Πυραύλων Ναυτικού στο Point Mugu στην ακτή της Καλιφόρνιας βόρεια του Λος Άντζελες ήταν ο John McLeod, ο οποίος παρέλαβε ένα νέο αναλογικό υπολογιστή το 1952. Ο John δεν ήταν ο μόνος μηχανικός στην αεροδιαστημική κοινότητα της Νότιας Καλιφόρνιας που αντιμετώπιζαν τα ίδια προβλήματα, και μάλιστα, ορισμένοι από αυτούς αποφάσισαν να δημιουργήσουν μια άτυπη ομάδα χρηστών για να ανταλλάξουν ιδέες και εμπειρίες πάνω στο συγκεκριμένο θέμα.

Η προσομοίωση των υπολογιστών δεν ήταν χρήσιμο εργαλείο στη δεκαετία του 1950. Χρειάστηκε πολύς χρόνος για να υπάρξουν αποτελέσματα, καθώς επίσης και πάρα πολλοί εξειδικευμένοι άνθρωποι, και συνεπώς κόστιζε σημαντικό ποσό τόσο στο προσωπικό όσο και στον υπολογιστή. Ένα παράδειγμα είναι η προσπάθεια

μοντελοποίησης των δεδομένων σε τηλεφωνικά συστήματα σε περιόδους αιχμής. Αυτό οφείλεται στο γεγονός ότι το σύστημα δεν συμμορφώθηκε με την μέθοδο αναμονής που χρησιμοποιήθηκε κατά τη διάρκεια αυτών των ημερών. Μία τεχνική που χρησιμοποιήθηκε ήταν η προσομοίωση γεγονότων με την βοήθεια του υπολογιστή. Τα εργαλεία που ήταν διαθέσιμα για την προσέγγιση ήταν ένα IBM 650, μια γλώσσα συναρμολόγησης, και μια ομάδα που απαρτιζόταν από έναν μαθηματικό, έναν μηχανικό συστημάτων και έναν προγραμματιστή. Η ομάδα πέτυχε λιγότερα από τα μισά από αυτά που έπρεπε να κάνουν, πήρε δύο φορές περισσότερο χρόνο και υπερπληθούσε τον προϋπολογισμό κατά δύο φορές.

Τον Οκτώβριο του 1961 η IBM παρουσίασε το "Gordon Simulator" στη Norden. Τον Δεκέμβριο του 1961 ο Geoffrey Gordon παρουσίασε το χαρτί του στο συνέδριο Joint Computer Conference για έναν προσομοιωτή συστημάτων γενικού σκοπού (GPSS). Αυτό το νέο εργαλείο χρησιμοποιήθηκε για να σχεδιαστεί το σύστημα για την FAA έτσι ώστε να διανέμει πληροφορίες για τις καιρικές συνθήκες στη αεροπορία. Η IBM παρείχε το λογισμικό και το υλικό. Η συγκεκριμένη εταιρεία κατάφερε να κατασκευάσει το μοντέλο, να προσομοιώσει το πρόβλημα και να λάβει απαντήσεις σε μόλις έξι εβδομάδες. Ένα νέο εργαλείο διατέθηκε για τους σχεδιαστές συστημάτων. Με την επιτυχία αυτού του εργαλείου άρχισαν να παράγονται μοντέλα για τις εξωτερικές ομάδες από το Norden και δημιουργήθηκε μια δραστηριότητα προσομοίωσης. Οι ομάδες πρόωρης προσομοίωσης δημιουργήθηκαν σε: Boeing, Martin Marietta, Διοίκηση Εφοδιαστικής Αεροπορίας, General Dynamics, Αεροσκάφη Hughes, Raytheon, Celanese, Exxon, Southern Railway και οι κατασκευαστές υπολογιστών ήταν IBM, Control Data, National Cash Register και UNIVAC.

Εν τω μεταξύ, στην εταιρεία Rand, οι Harry Markowitz, Bernard Hausner και Herbert Karr παρήγαγαν μια έκδοση του SIMSCRIPT το 1962 για να προσομοιώσουν τα προβλήματα διαχείρισης της αποθήκης τους. Στην Αγγλία οι J. Buxton και J. Laski ανέπτυξαν την CSL, μια γλώσσα ελέγχου και προσομοίωσης. Μια πρόωρη έκδοση του SIMULA αναπτύχθηκε στη Νορβηγία από τους O. Dahl και K. Nygaard και Don Knuth και J. McNeley παρήγαγαν SOL-A συμβολική γλώσσα για προσομοίωση γενικού σκοπού. Ο Ken Tocker έγραψε ένα σύντομο βιβλίο για την τέχνη της προσομοίωσης.

Το 1969, το τρίτο συνέδριο για την εφαρμογή προσομοίωσης πραγματοποιήθηκε τον Δεκέμβριο στο Λος Άντζελες. Ένα σημάδι της καθιέρωσης είναι ότι οι ΑΙΠΕ και TIMS εντάχθηκαν ως χορηγοί. Μεταξύ των νέων αντικειμένων ήταν η GASP και μια σύνοδος για τα συστήματα υγείας. Το τέταρτο και πέμπτο συνέδριο του 1970 και του 1971 πραγματοποιήθηκαν για τελευταία φορά στη Νέα Υόρκη. Στο τέταρτο συνέδριο συζητήθηκε το πρώτο εκπαιδευτικό πρόγραμμα GPSS του Tom Schriber. Στο πέμπτο συνέδριο έγινε ο πρώτος τίτλος της συνεδρίασης νυχτερινής προσομοίωσης. Ο αριθμός των σεμιναρίων αυξήθηκε με τον Alan Pritsker που καλύπτει τα GASP 2 ΚΑΙ Yen Chao SIMSCRIPT. Προστέθηκε μια εκπαιδευτική συνεδρία, αφού πολλά σχολεία προσέφεραν μαθήματα στην προσομοίωση συνεχών και διακριτών γεγονότων. Το πρώτο σεμινάριο SIMSCRIPT του Ed Russell δημοσιεύθηκε το 1976. Στη διάσκεψη του 1977 που πραγματοποιήθηκε στην Ουάσινγκτον, προστέθηκαν δύο νέες ενισχυτικές συνεδρίες για γεωργικά και στρατιωτικά συστήματα. Υπήρξε επίσης αυξημένο ενδιαφέρον για την εσωτερική λειτουργία της γλώσσας. Ένα παράδειγμα ήταν ένας βελτιωμένος κατάλογος των γεγονότων ALGORITHM που παρουσίασε ο Jim Henriksen.

Η προσομοίωση ήταν ένα θέμα που διδάχθηκε στους βιομηχανικούς μηχανικούς στο σχολείο αλλά σπάνια εφαρμοζόταν. Οι πολλές ώρες που περνούσαν οι μαθητές στο τερματικό του υπολογιστή και οι φαινομενικά ατελείωτες επαναλήψεις για να βρουν ένα σφάλμα σε μια γλώσσα ήταν αυτό που σήμαινε «προσομοίωση» για τους απόφοιτους στο Ι.Ε. στη δεκαετία του '70. Όταν τα εργαλεία υπολογιστικών φύλλων εισήχθησαν για πρώτη φορά στα τέλη της δεκαετίας του '70, χρησιμοποιήθηκαν μόνο από «λίγους πιστούς». Η δημοτικότητα της προσομοίωσης ως ισχυρό εργαλείο αυξήθηκε με τον αριθμό των διασκέψεων και συνεδριών. Ο αριθμός των συνεδριάσεων που πραγματοποιήθηκαν πάνω στην προσομοίωση διπλασιάστηκε το 1971 και συνέχισε να αυξάνεται σε περίπου σαράντα συνεδρίες το 1977 και εξήντα συνεδρίες το 1983 σε σύγκριση με τις 12 συνεδρίες του 1967. Ένα σημάδι της αυξανόμενης ωριμότητας στον τομέα αυτό ήταν μια Panel Discussion στο Μαϊάμι το 1978 στα «ατυχήματα της προσομοίωσης», εστιάζοντας στο τι μπορεί και τι γίνεται λάθος και σε ένα έγγραφο σχετικά με τα Έργα ΔΙΑΧΕΙΡΙΣΗΣ ΠΡΟΣΟΜΟΙΩΣΗΣ. Το 1979 η διάσκεψη πραγματοποιήθηκε στο Σαν Ντιέγκο και το 1980 διεξήχθη στο Ορλάντο. Υπήρχαν

περισσότερα μαθήματα και ημερίδες οργανώθηκαν σε εκθέσεις συνεδριών για αρχάριους, ενδιάμεσους και προηγμένους χειριστές.

Το 1982, τα περισσότερα λογισμικά προσομοίωσης επικεντρώθηκαν στον προγραμματισμό των υλικών απαιτήσεων (MRP), ο οποίος εξετάζει μόνο το χρονοδιάγραμμα και το μέγεθος των παραγγελιών χωρίς να λαμβάνεται υπόψη τον περιορισμό της παραγωγικής ικανότητας. Το λογισμικό δεν αναπτύχθηκε περαιτέρω, για να δώσει μια πραγματική σημασία ενός αυτοματοποιημένου εργοστασίου. Εκατοντάδες ρομπότ και εξοπλισμοί αξίας εκατομμυρίων δολαρίων εξοπλισμού ήταν άχρηστα, καθώς δεν χρησιμοποιήθηκαν επαρκώς και χρησιμοποιήθηκαν για λάθος σκοπό εξαιτίας του κακού σχεδιασμού. Το 1982 οι προσωπικοί μικροϋπολογιστές ήταν 16-bit μηχανές ικανές να κρατούν μνήμες της τάξης των 128kb, 256kb, ή ακόμα και 512kb. Δεν υπήρχε διαθέσιμο λογισμικό το οποίο μπορούσε να εκμεταλλευτεί τον μικροεπεξεργαστή 16bit και την πρόσθετη μνήμη. Το 1983 ο αριθμός των εταιρειών που χρησιμοποιούν προσομοίωση ήταν μικρός. Με την εξέλιξη των πληροφοριακών συστημάτων τα οποία μπορούν να συλλέξουν και να αποθηκεύσουν πολλά δεδομένα για την κατασκευή και τη συντήρηση μοντέλων, η προσομοίωση για τον προγραμματισμό παραγωγής έγινε εφικτή. Το ευρέως χρησιμοποιούμενο σύστημα διαχείρισης εργοστασίου από την CAM-I υποστηρίζει και διανέμει, τον έλεγχο των λειτουργιών του συνεργείου και τις επικοινωνίες μεταξύ αλγορίθμων σχεδιασμού και λειτουργίας. Κατά την εγκατάσταση ενός τέτοιου συστήματος, μεγάλο μέρος των προβλημάτων που σχετίζονται με την κατασκευή και συντήρηση μοντέλων προσομοίωσης εξαλείφθηκε.

Το 1984 αναπτύχθηκε η πρώτη γλώσσα προσομοίωσης ειδικά σχεδιασμένη για συστήματα παραγωγής μοντέλων. Στα τέλη της δεκαετίας του '80 με την ανάπτυξη του μοντέλου προσομοίωσης διακριτών συμβάντων, η διοίκηση ήταν σε θέση να αξιολογήσει το κόστος-οφέλος των εναλλακτικών λύσεων, τις στρατηγικές συντήρησης, την επισκευή εξοπλισμού μετατροπής και την αντικατάσταση κεφαλαίων.

Στις αρχές της δεκαετίας του '90 άρχισε να εμφανίζεται λογισμικό όπως το EMS version of GPSS / PC, το οποίο επέτρεψε στους χρήστες των συμβατών προσωπικών υπολογιστών της IBM να έχουν πρόσβαση σε πρόσθετη μνήμη, πάνω από το όριο των 640k που επέβαλε η αρχική αρχιτεκτονική PC. Το EXTEND ήταν μια εφαρμογή

προσομοίωσης γραφικής παράστασης με βάση το Macintosh που υποστηρίζει τόσο τη διακριτική όσο και τη συνεχή προσομοίωση συμβάντος. Η έκδοση MIC-SIM 3.0 παρείχε δυνατότητες μοντελοποίησης και χαρακτηριστικά που ήταν τόσο εύκολο να μάθουν και να χρησιμοποιούν αυτές τις υπηρεσίες κατάρτισης και συμβουλευτικής δεν χρειάζονται πλέον. Το GPSS / Η υποστηρίχθηκε από ένα ευρύ φάσμα υλικού στον κλάδο, από τους υπολογιστές και τους περισσότερους σταθμούς εργασίας Unix έως τα συστήματα VAX / VMS και mainframe της IBM. Προσέφερε πολυάριθμες επεκτάσεις, οι οποίες εμπόδισαν τους χρήστες να γράψουν εξωτερικό κώδικα στο Fortran ή στο «C». Το MAST παρείχε ένα ενιαίο περιβάλλον για το σχεδιασμό, την απόκτηση και τη λειτουργία των συστημάτων παραγωγής. Δεν απαιτούσε προγραμματισμό, ούτε μοντελοποίηση, ούτε καν απαιτείται επεξεργασία κειμένου για τη μελέτη ενός συστήματος παραγωγής.

Η δύναμη της προσομοίωσης ως εργαλείου έγινε εμφανής στα μέσα της δεκαετίας του 90. Οι αιτίες αντιμετώπισαν εταιρείες όπως η Universal Data Systems (υπερσύγχρονη μονάδα συναρμολόγησης ηλεκτρονικών). Το εμπόδιο ήταν να μετατρέψουμε ολόκληρο το εργοστάσιο σε ένα υβριδικό κατάστημα ροής όπου μια μεμονωμένη μονάδα θα αποστέλλεται στην επόμενη λειτουργία μόλις ολοκληρωθεί στην τρέχουσα λειτουργία. Μια σοβαρή επιφύλαξη για αυτή την αλλαγή ήταν η επίδραση στην απογραφή τελικών προϊόντων. Πειράματα πραγματοποιήθηκαν χρησιμοποιώντας το πρόγραμμα προσομοίωσης που γράφτηκε στο GPSS / PC (Minuteman) χρησιμοποιώντας ένα PC / AT IBM. Το σύνολο του προγράμματος χρειάστηκε 30 ημέρες για την προσομοίωση και τα αποτελέσματα ήταν θετικά με την ενδεχόμενη μετατροπή ολόκληρου του εργοστασίου σε περιβάλλον ροής καταστημάτων σε σύγκριση με το αρχικό περιβάλλον παρτίδας.

Το 1998 άρχισε να ξεχωρίζει λογισμικό όπως το Micro Saint version 2.0 για τα Windows 95. Παρείχε αυτόματη συλλογή δεδομένων, βελτιστοποίηση και νέα διεπαφή των Windows. Εκτός από αυτό, δεν απαιτεί την ικανότητα να γράφει σε οποιαδήποτε γλώσσα προγραμματισμού. Σήμερα, η προσομοίωση έχει προχωρήσει σε τέτοιο στάδιο ώστε το λογισμικό να επιτρέπει στο χρήστη να μοντελοποιεί, να εκτελεί και να ζωντανεύει οποιοδήποτε σύστημα παραγωγής σε οποιοδήποτε επίπεδο λεπτομέρειας. Ένας σύνθετος μεταφορέας μήκους 2000 ποδών μπορεί να μοντελοποιηθεί σε λίγα

λεπτά. Τα προϊόντα, ο εξοπλισμός και οι πληροφορίες αντιπροσωπεύονται από μια ενιαία οντότητα που συνδέεται με τέσσερις διαστάσεις (X, Y, Z και χρόνο) και έναν ορισμό της συμπεριφοράς του.

1.3 ΣΥΣΤΗΜΑ

Μια πολύ βασική έννοια είναι το σύστημα. Το σύστημα με βάση τους ορισμούς που δόθηκαν από τους Δ. Καλλιγερόπουλος και Σ. Βασιλειάδου (2005) είναι ένα σύνολο στοιχείων που σχηματίζουν μια ενότητα και δρουν σαν μια ενότητα, το σύστημα καθορίζεται από **2 χαρακτηριστικά** και από τα **μεγέθη επικοινωνίας**:

- Το πρώτο χαρακτηριστικό είναι η δομή που αποτελεί την εσωτερική διάταξη των στοιχείων του
- Το δεύτερο χαρακτηριστικό είναι η δυναμική λειτουργία και καθορίζει τη σχέση που συνδέει τα στοιχεία μεταξύ τους αλλά και με το περιβάλλον του συστήματος
- Τα μεγέθη επικοινωνίας είναι η είσοδος και η έξοδος του συστήματος, η είσοδος είναι η επίδραση που δέχεται το σύστημα από το περιβάλλον ενώ η έξοδος είναι το αποτέλεσμα που απευθύνει το σύστημα προς το περιβάλλον.

Όπως ανέφεραν οι Δ. Καλλιγερόπουλος και Σ. Βασιλειάδου (2005), ένα σύστημα που αποτελείται από παραπάνω από μια εισόδους ή εξόδους ονομάζεται πολλαπλό σύστημα και ανάλογα με τον αριθμό εισόδων και εξόδων χωρίζεται σε **4 κατηγορίες**, που είναι:

- **Μονής εισόδου-μονής εξόδου** (Single Input-Single Output,SISO)
- **Μονής εισόδου-πολλαπλής εξόδου** (Single Input-Multiple Output,SIMO)
- **Πολλαπλής εισόδου-μονής εξόδου** (Multiple Input-Single Output,MISO)
- **Πολλαπλή εισόδου-πολλαπλής εξόδου** (Multiple Input-Multiple Output,MIMO)

1.3.1 ΚΑΤΗΓΟΡΙΕΣ ΣΥΣΤΗΜΑΤΩΝ

Ο Μαλατέστας Β. Παντελής (2011) αναφέρει ότι τα συστήματα αναλόγως με τη σύνθεση τους κατηγοριοποιούνται σε:

- **Δυναμικά-στατικά συστήματα.**

Δυναμικά είναι τα συστήματα που λόγω διέγερσης της εισόδου παρουσιάζουν μεταβολή στην έξοδο μετά από κάποια χρονική καθυστέρηση και περιγράφονται από εξισώσεις που περιέχουν ολοκληρώματα και διαφορίση. Ενώ τα **στατικά** συστήματα είναι συστήματα που οι έξοδοι τους εξαρτώνται αποκλειστικά από τις εισόδους τους την ίδια χρονική στιγμή και περιγράφονται από αλγεβρικές εξισώσεις. Τα δυναμικά συστήματα είναι και συστήματα με μνήμη διότι η έξοδος τους εξαρτάται και από τις προηγούμενες τιμές που είχε η είσοδος τους σε αντίθεση με τα στατικά συστήματα που η έξοδος τους εξαρτάται μόνο από τις τιμές που παίρνει η είσοδος τους μια συγκεκριμένη χρονική στιγμή γι' αυτό τα στατικά συστήματα ονομάζονται συστήματα χωρίς μνήμη. Συνήθως αν ένα σύστημα έχει τουλάχιστον ένα δυναμικό στοιχείο τότε το σύστημα χαρακτηρίζεται ως δυναμικό. Σύστημα $2(t) \ x_1(t) \dots y_1(t) \ y_2(t) \ x_n(t) \ y_n(t) \dots 5$

- **Γραμμικά και μη γραμμικά συστήματα.**

Γραμμικά συστήματα, είναι τα συστήματα στα οποία ισχύει η αρχή της υπέρθεσης (principle of superposition). Η αρχή της υπέρθεσης αναφέρει ότι η συνολική απόκριση ενός συστήματος ισούται με τις επιμέρους αποκρίσεις της κάθε μιας εισόδου ξεχωριστά παίρνοντας κάθε φορά τις υπόλοιπες εισόδους να είναι ίσες με τον μηδέν. Ενώ **μη γραμμικά** συστήματα, είναι τα συστήματα στα οποία δεν ισχύει η αρχή της υπέρθεσης.

- **Χρονικά μεταβλητά και χρονικά αμετάβλητα συστήματα.**

Χρονικά μεταβλητά συστήματα, είναι τα συστήματα τα οποία η μορφή της εξόδου εξαρτάται αποκλειστικά από τη μορφή της εισόδου για συγκεκριμένο χρονικό διάστημα που εφαρμόζεται η είσοδος στην έξοδο. Ενώ τα **χρονικά αμετάβλητα** συστήματα, είναι τα συστήματα που η μορφή της εξόδου εξαρτάται αποκλειστικά από τη μορφή της εισόδου αλλά χωρίς να υπάρχει συγκεκριμένο χρονικό διάστημα για την εφαρμογή της εισόδου.

- **Συνεχούς χρόνου και διακριτού χρόνου συστήματα.**

Οι Averill M. Law και W. David Kelton το 1991 ανέφεραν τους ορισμούς για τα συστήματα διακριτού χρόνου και συνεχούς χρόνου. Στο σύστημα **διακριτού**

χρόνου, οι μεταβλητές κατάστασης αλλάζουν στιγμιαία σε ξεχωριστά χρονικά διαστήματα ενώ στο σύστημα **συνεχούς χρόνου**, οι μεταβλητές κατάστασης αλλάζουν συνεχώς σε σχέση με το χρόνο.

- **Αιτιατά και μη αιτιατά συστήματα.**

Αιτιατά συστήματα είναι τα συστήματα που η έξοδος του εξαρτάται από παραπάνω από μια τιμή της εισόδου σε συγκεκριμένη χρονική στιγμή αλλά και από τις προηγούμενες τιμές της εισόδου. Από την άλλη πλευρά **μη αιτιατά συστήματα** είναι αυτά που η έξοδος εξαρτάται από τις μελλοντικές τιμές της εισόδου του συστήματος.

- **Ντετερμινιστικά και στοχαστικά συστήματα.**

Τα **ντετερμινιστικά** συστήματα είναι αυτά τα συστήματα στα οποία γνωρίζουμε τις εισόδους τους ενώ στα **στοχαστικά** συστήματα οι εισοδοί είναι άγνωστες.

- **Συστήματα συγκεντρωμένων και κατανεμημένων παραμέτρων.**

Τα συστήματα **συγκεντρωμένων παραμέτρων**, είναι τα συστήματα που έστω μια παράμετρος δεν είναι διανεμημένη σε μια μεταβλητή(εκτός του χρόνου) ενώ τα συστήματα **κατανεμημένων παραμέτρων**, είναι τα συστήματα που ακόμη και μια παράμετρος είναι διανεμημένη σε κάποια μεταβλητή εκτός του χρόνου.

1.4 MONTELO

Όπως αναφέρει ο B. Wayne Bequette (1998), μοντέλο είναι μια μικρή αναπαράσταση ενός σχεδιαζόμενου ή υπάρχοντος αντικειμένου. Τα μοντέλα αποτελούνται από εξισώσεις:

- **Αλγεβρικές**
- **Διαφορικές**
- **Μερικής διαφορίσης**

Για την εύρεση του μοντέλου κάνουμε διάφορα πειράματα, δηλαδή διεγείρετε/διεγείρονται την/τις είσοδο/εισόδους ενός συστήματος και παρατηρούνται τα αποτελέσματα στην/στις έξοδο/εξόδους του συστήματος.

1.4.1 ΚΑΤΗΓΟΡΙΕΣ ΜΟΝΤΕΛΩΝ

Ο Dan Neculescu ανέφερε ότι τα μοντέλα χωρίζονται σε **3 κατηγορίες**:

1. **Μοντέλα συνεχούς χρόνου**(continuous time models)

Τα μοντέλα αυτά περιγράφονται από διαφορικές εξισώσεις.

2. **Μοντέλα διακριτού χρόνου**(discrete time models)

Σε αυτά τα μοντέλα ο χρόνος είναι χωρισμένος σε ίσα διαστήματα και περιγράφονται και αυτά με διαφορικές εξισώσεις.

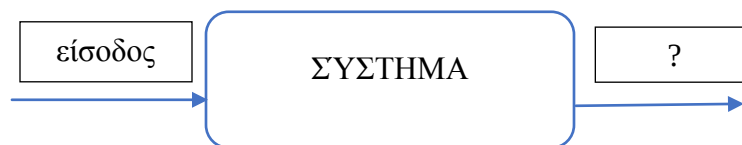
3. **Μοντέλα διακριτών συμβάντων**(discrete events models)

Τα μοντέλα αυτά αναπαριστούν μεταβολές μεταβλητών κατάστασης που γίνονται σε διαφορετικές χρονικές στιγμές.

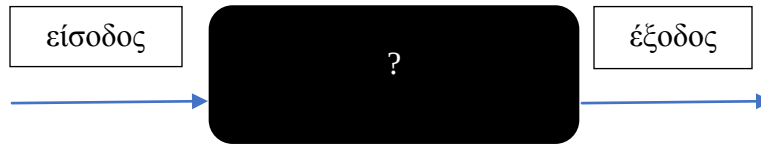
1.4.2 ΤΕΧΝΙΚΕΣ ΕΥΡΕΣΗΣ ΜΟΝΤΕΛΟΥ

Όμως μπορεί να έχουμε κάποια προβλήματα για την εύρεση του μοντέλου, για παράδειγμα να μην γνωρίζουμε τις εισόδους ή τις εξόδους ή το ίδιο το σύστημα ή κάποιο συνδυασμό αυτών. Γι' αυτό οι Mohd. Ehmer Khan και Farmeena Khan το 2012 ανέλυσαν **3 τεχνικές**:

- 1^η είναι η τεχνική «λευκό κουτί», στο οποίο η είσοδος και το σύστημα είναι γνωστά, και αυτό που αναζητείται η έξοδος, επομένως με βάση τις πληροφορίες που συλλέγονται από το σύστημα και την είσοδο, μπορεί να βρεθεί η έξοδος.



- 2^η είναι η τεχνική «μαύρο κουτί», στο οποίο η είσοδος και η έξοδος είναι γνωστά, και ως ζητούμενο είναι το σύστημα, άρα οι πληροφορίες που συλλέγονται από την είσοδο και την έξοδο συμβάλουν στο να εντοπιστεί το σύστημα.



- 3^η είναι η τεχνική «το γκρι κουτί», στο οποίο είναι γνωστά το σύστημα και η έξοδος και είναι άγνωστη η είσοδος του συστήματος, άρα συλλέγοντας πληροφορίες από την έξοδο και το σύστημα μπορεί να βρεθεί η είσοδος.



1.4.3 ΚΑΤΑΣΚΕΥΗ ΜΟΝΤΕΛΩΝ ΠΡΟΣΟΜΟΙΩΣΗΣ

Η κατασκευή μοντέλων προσομοίωσης είναι μια δύσκολη τεχνική γιατί πρέπει να εξισορροπήσει αντικρουόμενους παράγοντες. Αφενός το μοντέλο θα πρέπει να είναι αρκετά απλό, έτσι ώστε να μπορεί να κατασκευασθεί και να μελετηθεί, αφετέρου θα πρέπει να είναι αρκετά πολύπλοκο, έτσι ώστε να αντιπροσωπεύει όσο πιο πιστά γίνεται το σύστημα που πρόκειται να μελετηθεί. Η ισορροπία αυτή μπορεί να επιτευχθεί με προσεκτική ανάλυση του μοντέλου. Αν το μοντέλο που έχει κατασκευασθεί είναι απλό και κατανοητό, τότε μπορεί να εμπλουτισθεί. Αντίθετα, αν το μοντέλο είναι πολύπλοκο και δυσνόητο, μπορεί να απλοποιηθεί.

Σε κάθε περίπτωση χρησιμοποιούνται οι εξής πέντε κανόνες εμπλουτισμού ή απλοποίησης του μοντέλου:

- Αν το μοντέλο είναι απλό, ορισμένες σταθερές μετατρέπονται σε μεταβλητές ώστε να γίνει πολύπλοκο. Αντίθετα αν το μοντέλο είναι πολύπλοκο, ορισμένες μεταβλητές μετατρέπονται σε σταθερές.
- Η εξάλειψη ή συνένωση μεταβλητών απλοποιεί το μοντέλο, ενώ αντίθετα η προσθήκη μεταβλητών το εμπλουτίζει.

- Επειδή τα γραμμικά μοντέλα είναι γενικώς απλούστερα των μη γραμμικών, η παραδοχή γραμμικότητάς του συστήματος απλοποιεί το μοντέλο, ενώ αντίθετα η παραδοχή μη γραμμικότητάς το εμπλουτίζει.
- Η προσθήκη ισχυρότερων υποθέσεων και περιορισμών απλοποιεί το μοντέλο.
- Τέλος, ο περιορισμός των ορίων του συστήματος οδηγεί σε απλούστερο μοντέλο, ενώ η επέκτασή τους οδηγεί σε πιο πολύπλοκο μοντέλο.

1.5 ΕΦΑΡΜΟΓΕΣ ΤΗΣ ΠΡΟΣΟΜΟΙΩΣΗΣ

Η προσομοίωση έχει χρησιμοποιηθεί σε ευρύ πεδίο περιοχών, μερικές από τις οποίες θα αναφέρουμε στις επόμενες παραγράφους. Ορισμένες από τις περιοχές εφαρμογής της προσομοίωσης στο χώρο παροχής υπηρεσιών και οι αντίστοιχοι στόχοι ανάλυσης και σχεδιασμού, είναι οι ακόλουθοι:

- **Επικοινωνίες**

Η εφαρμογή της προσομοίωσης είναι ιδιαίτερα ζωτική στο χώρο των επικοινωνιών. Τοπικά δίκτυα υπολογιστών και δίκτυα ευρείας ζώνης, τηλεφωνικά συστήματα, εθνικά και διεπαιρωτικά συστήματα δορυφορικών επικοινωνιών, δίκτυα καλωδιακής τηλεόρασης και συστήματα κινητής τηλεφωνίας είναι μερικά από τα παραδείγματα πολύπλοκων συστημάτων που απαιτούν τη συμβολή της προσομοίωσης με υπολογιστή για αποδοτική σχεδίαση και λειτουργία. Η εμφάνιση των λεωφόρων επικοινωνίας και η άνθισή τους στο τέλος της δεκαετίας του 90 δημιουργεί ακόμα μεγαλύτερη ζήτηση για μελέτες με χρήση προσομοίωσης. Για την ανάλυση και το σχεδιασμό επικοινωνιακών συστημάτων διατίθενται ήδη αρκετά εμπορικά εργαλεία προσομοίωσης με ειδικούς σκοπούς.

- **Οικονομία**

Υπάρχουν πολλές αναφορές σχετικά με την εφαρμογή της προσομοίωσης σε τράπεζες, εταιρείες ασφαλειών και προστασίας. Μερικές από τις δραστηριότητες που μπορούν να εκτιμηθούν με την προσομοίωση είναι: η ανάλυση των συναλλαγών και της ροής χρημάτων, ο σχεδιασμός συστημάτων γραφείων, ο προγραμματισμός υλικών και προμηθειών, ο σχεδιασμός επεξεργασίας

δεδομένων και δικτύων υπολογιστών, ο σχεδιασμός αυτόματων ταμειακών μηχανών και συστημάτων εξυπηρέτησης.

- **Υπηρεσίες τροφίμων**

Συστήματα όπως διάφορα είδη εστιατορίων και συστήματα αποθήκευσης τροφίμων μπορούν να γίνουν αντικείμενα μελετών προσομοίωσης με σκοπούς όπως η διαχείριση της αποθήκης των υλικών και ο σχεδιασμός προμηθειών, ο σχεδιασμός διανομής, η επιλογή θέσεων και ο σχεδιασμός εργατικού δυναμικού.

- **Υγεία**

Νοσοκομεία, μονάδες εντατικής θεραπείας, ιατρεία, οδοντιατρεία και παραϊατρικά γραφεία μελετώνται συχνά μέσω της προσομοίωσης για να καθοριστούν οι βάρδιες για τις νοσοκόμες και τους γιατρούς, οι πολιτικές για τη διαχείριση των αποθεμάτων φαρμάκων και τροφίμων και τα σχέδια τα σχετικά με τη διαθεσιμότητα σε πόρους όπως κρεβάτια, χώρους αναμονής, χειρουργεία, εξοπλισμούς εργαστηρίων και ασθενοφόρα. Επιπλέον, μελέτες επιδημιών όπως οι προβλεπόμενοι ρυθμοί εξάπλωσης μιας ασθένειας ή ανάλυση εναλλακτικών πολιτικών ελέγχου ασθενειών, θεωρούνται ως εργασίες ρουτίνας για την προσομοίωση.

- **Υπηρεσίες φιλοξενίας και ξενοδοχείων**

Συστήματα όπως ξενοδοχεία, πανδοχεία και θέρετρα μπορούν να μελετηθούν με χρήση προσομοίωσης για να καθοριστούν παράγοντες όπως: κατάλληλες δυναμικότητες, περιοχές, πολιτικές διαχείρισης αποθηκών και πόρων, μέθοδοι σχεδιασμού και προγραμματισμού ανθρώπινου δυναμικού και συστήματα κρατήσεων.

- **Μεταφορές**

Εδώ τα συστήματα περιέχουν έναν ή περισσότερους τύπους οχημάτων (π.χ. ταξί, λεωφορεία, τρένα αεροπλάνα, πλοία), επιβάτες, φορτία και διαδρομές μεταφοράς. Η μελέτη προσομοίωσης μπορεί να έχει ως στόχο της τον καθορισμό της χωρητικότητας των οχημάτων το εργατικό δυναμικό (οδηγούς, προσωπικό συντήρησης κ.λ.π.), το σχεδιασμό αποθεμάτων ανταλλακτικών, το σχεδιασμό συντήρησης, τη δρομολόγηση των οχημάτων, το σχεδιασμό λεωφόρων, το

σχεδιασμό συστημάτων ελέγχου κίνησης αέρα και εδάφους και το σχεδιασμό των δομών και των χώρων στάθμευσης.

- **Πρόγνωση καιρού, περιβάλλον και οικολογία**

Για την πρόγνωση του καιρού χρησιμοποιείται μόνιμα και απαρέγκλιτα η προσομοίωση με υπολογιστή. Μεγάλος αριθμός μεταβλητών διαχειρίζονται από προγράμματα προσομοίωσης, που συνήθως εκτελούνται σε υπερ-υπολογιστές, για την πρόβλεψη τοπικών και παγκόσμιων καιρικών συνθηκών σε διάφορες χρονικές στιγμές. Οι μελέτες που αφορούν τον έλεγχο της ρύπανσης, το φαινόμενο του θερμοκηπίου, τους πληθυσμούς των εντόμων και άλλα περιβαλλοντικά και οικολογικά ζητήματα πραγματοποιούνται επίσης με χρήση προσομοίωσης με υπολογιστή. Τα συστήματα παραγωγής και κατασκευών είναι μια ακόμα σημαντική ενότητα για εφαρμογές προσομοίωσης.

- **Εξόρυξη φυσικών πόρων**

Βιομηχανικές μονάδες στον τομέα των ορυχείων, της υλοτομίας, της άντλησης υδάτων και της αλιείας χρησιμοποιούν προσομοίωση με υπολογιστή στη σχεδίαση σχετικών δραστηριοτήτων και τη δημιουργία πολιτικών για έγκαιρη προκήρυξη διαγωνισμών για την αγορά ακριβών μέσων όπως μεγάλων τρακτέρ, φορτωτών, εκσκαφέων, ανυψωτών, γερανών, μεταφορέων, προωθητών γαιών και πλοίων.

- **Παραγωγή ενέργειας**

Τα συστήματα παραγωγής ενέργειας τα οποία βασίζονται σε πηγές όπως ο ατμός, τα καύσιμα, το νερό, τα πυρηνικά υλικά, ο ήλιος ή ο άνεμος συνήθως προσομοιώνονται για το σχεδιασμό συστημάτων δυναμικότητας, σύνθεση και διανομής και για την ανάλυση και το σχεδιασμό λειτουργικών συστημάτων σχετικά με το προγραμματισμό ρυθμών παραγωγής, το προγραμματισμό διανομής, το σχεδιασμό συστημάτων ελέγχου, το σχεδιασμό συστημάτων ασφάλειας και αξιοπιστίας, τον προγραμματισμό συντήρησης και τον έλεγχο των περιβαλλοντικών επιπτώσεων.

- **Βιομηχανίες**

Κάθε είδος βιομηχανίας, συμπεριλαμβανομένων των εργοστασίων επεξεργασίας χημικών, των αυτοκινητοβιομηχανιών, των βιομηχανιών κατασκευής

αεροπλάνων, μηχανημάτων, ηλεκτρονικών συσκευών κ.λ.π. χρησιμοποιεί εκτεταμένα την προσομοίωση σε εφαρμογές όπως ο στρατηγικός σχεδιασμός, η μέση παραγωγική ικανότητα και ο σχεδιασμός παραγωγής, η μελέτη κατασκευής εργοστασίου, ο σχεδιασμός πολιτικής επιλογής εξοπλισμού, αντικατάστασης και συντήρησης, ελέγχου και προγραμματισμού αποθεμάτων, προγραμματισμού παραγωγής, αποθήκευσης και διαχείρισης υλικών, βελτίωσης γραμμής παραγωγής και πολλά άλλα ζητήματα που σχετίζονται με το σχεδιασμό, την κατασκευή, τη συναρμολόγηση, τον έλεγχο ποιότητας, την αποθήκευση και τη διανομή. Η ζήτηση των μελετών προσομοίωσης στα βιομηχανικά συστήματα αυξάνεται συνέχεια. Κατά συνέπεια υπάρχει ήδη ένας σημαντικός αριθμός εργαλείων προσομοίωσης ειδικού σκοπού, που προσφέρονται ως εμπορικά προϊόντα για σχεδιασμό και ανάλυση βιομηχανικών συστημάτων.

ΚΕΦΑΛΑΙΟ 2: ΜΟΝΤΕΛΑ ΕΠΙΔΗΜΙΟΛΟΓΙΑΣ

2.1 Η ΕΝΝΟΙΑ ΤΗΣ ΕΠΙΔΗΜΙΟΛΟΓΙΑΣ

Αρχικά, ας προσπαθήσουμε να ορίσουμε την επιδημιολογία. **Επιδημιολογία**³ είναι η μελέτη της κατανομής και της εξέλιξης διαφόρων νοσημάτων στον ανθρώπινο πληθυσμό (περιγραφική επιδημιολογία) και των παραγόντων που τις διαμορφώνουν ή μπορούν να τις επηρεάσουν (αναλυτική επιδημιολογία). Ο ενεργός επιδημιολόγος που εργάζεται σε τομείς η θεματολογία των οποίων εκτείνεται από πρακτικά ζητήματα, όπως η διερεύνηση μίας επιδημίας, μία περιβαλλοντική έκθεση και η αγωγή υγείας, μέχρι πιο θεωρητικά, όπως η ανάπτυξη στατιστικών, μαθηματικών, φιλοσοφικών, βιολογικών και ψυχοκοινωνικών θεωριών. Έτσι, οι επιδημιολόγοι ασχολούνται σε μία σειρά τύπων επιστημονικής μελέτης από μελέτες παρατήρησης έως πειραματικές μελέτες- με σκοπό τον προσδιορισμό αμερόληπτων σχέσεων μεταξύ εκθέσεων, όπως η διατροφή, οι βιολογικοί παράγοντες, το άγχος, και του αποτελέσματος, όπως π.χ. μία ασθένεια, η υγεία κ.τ.λ. Στη σύγχρονη επιδημιολογία, χρησιμοποιούνται μέθοδοι και τεχνικές από την Πληροφορική της Υγείας.

Ειδικότερα, οι επιδημίες είναι ξεσπάσματα μολυσματικών ή μεταδοτικών ασθενειών που έχουν επιπτώσεις σε μερικά μέλη ενός πληθυσμού που ζει σε ιδιαίτερες θέσεις σε συγκεκριμένους χρόνους. Είναι συνήθως βραχυπρόθεσμα, οξέα, συχνά βίαια γεγονότα που πάνε κι έρχονται. Επίσης μπορούν να είναι ιδιαίτερα καταστρεπτικές από άποψη απώλειας ζωής, αλλά και όσον αφορά τον αντίκτυπό τους στην οικονομία, στην οικογένεια, στην κοινωνία και στον πολιτισμό. Υπάρχουν ακόμη και παραδείγματα κρατών και κυβερνήσεων που κατέρρευσαν μετά από καταστρεπτικές επιδημίες. Τέλος οι επιδημίες μπορεί να είναι και συνέπεια καταστροφών άλλου είδους, όπως οι τροπικές θύελλες, πλημμύρες, σεισμοί, ξηρασίες, κ.λ.π. ενώ μπορούν επίσης να προσβάλουν τα ζώα, προκαλώντας έτσι τοπικές οικονομικές καταστροφές.

Οι πιο γνωστές επιδημικές ασθένειες διαδίδονται από την άμεση επαφή μεταξύ μολυσμένου και μη που είναι τα ευπαθή πρόσωπα. π.χ. ιλαρά, ευλογιά, γρίπη,

³ Epidemiology, https://en.wikipedia.org/wiki/Epidemiology#cite_note-Epidemiology_2009-38, 21 Μαρτίου 2005, τελευταία πρόσβαση 19/5/2019.

φυματίωση, πολιομυελίτιδα, VIH/AIDS, και σεξουαλικά μεταδιδόμενες μολύνσεις (STIs) όπως η σύφιλη. Μια σημαντική δεύτερη κατηγορία επιδημικών ασθενειών είναι αυτές που διαβιβάζονται στα ανθρώπινα όντα από έντομα. Μερικά παραδείγματα (και οι μεταφορείς τους): Βουβωνική πανούκλα (ψύλλος), ελονοσία (κουνούπι), τύφος (ψείρα σωμάτων) και κίτρινος πυρετός (κουνούπι). Μια τρίτη ομάδα περιλαμβάνει εκείνες τις επιδημικές ασθένειες που επικοινωνούν από το νερό- ή από τροφικά παθογόνα (βακτηρίδια, ιούς ή άλλους μικροοργανισμοί), π.χ. χολέρα, τυφοειδής και δυσεντερία. Δεν είναι όλες οι ασθένειες σε θέση να προκαλέσουν επιδημίες. Οι διάφορες μορφές καρκίνου και καρδιακών παθήσεων επηρεάζονται από τη γενετική προδιάθεση και του τρόπου ζωής (κάπνισμα, άσκηση, διατροφή). Η σημαντική διαφορά είναι ότι δεν μπορούν να επικοινωνήσουν από πρόσωπο με πρόσωπο. Αυτές είναι οι ενδημικές ασθένειες και βρίσκονται τακτικά σε έναν πληθυσμό.

Η επιδημιολογία έχει αποδειχθεί μεγίστης σημασίας στην προσπάθεια μελέτης των επιπτώσεων διαφόρων φυσικών και χημικών παραγόντων στην ανθρώπινη υγεία. Συγκεκριμένα, στην περίπτωση του καρκίνου η επιδημιολογία παίζει σημαντικό ρόλο στην προσπάθεια ποσοτικοποίησης του κινδύνου. Παρόλα αυτά, πρέπει να λαμβάνονται υπόψη ορισμένα στοιχεία στην προσπάθεια ερμηνείας των επιδημιολογικών αποτελεσμάτων, όπως αναφέρεται και στη συνέχεια.

Στόχοι της επιδημιολογίας είναι οι ακόλουθοι :

- Η μέτρηση και περιγραφή της νοσηρότητας και θνησιμότητας ενός πληθυσμού.
- Ο έλεγχος της χρονικής εξέλιξης μιας νόσου.
- Η ανακάλυψη αιτιολογικών παραγόντων που προκαλούν μία νόσο (επαγωγική Επιδημιολογία).
- Η μελέτη των συνθηκών και των αιτιών που προκαλούν επιδημίες (monitoring)· για παράδειγμα κάπνισμα και καρκίνος των πνευμόνων.
- Η κατανόηση της ιστορίας των νοσημάτων (για πρόληψη και έλεγχο-screening).
- Η ταξινόμηση των νοσημάτων.
- Ο προγραμματισμός, οργάνωση και αξιολόγηση των υπηρεσιών υγείας.

Συνεχίζοντας, οι επιδημιολόγοι χρησιμοποιούν μια σειρά από σχέδια μελέτης από την **παρατηρησιακή** έως την **πειραματική** επιδημιολογία, και γενικά κατηγοριοποιούνται ως:

- **Περιγραφικά:** Αφορούν το ποιος, τι, πού και πότε συμβαίνει κατάσταση της υγείας.
- **Αναλυτικά:** Έχουν στόχο την περαιτέρω εξέταση γνωστών συσχετίσεων ή υποθετικών σχέσεων) και αφορούν περισσότερο τον «τρόπο» με τον οποίο ένα συμβάν σχετίζεται με την υγεία.

Στις παρατηρητικές μελέτες, η φύση επιτρέπεται να «λάβει την πορεία της», όπως παρατηρούν οι επιδημιολόγοι. Αντίθετα, σε πειραματικές μελέτες, ο επιδημιολόγος είναι αυτός που ελέγχει όλους τους παράγοντες που εισάγουν μια συγκεκριμένη μελέτη περίπτωσης. Οι επιδημιολογικές μελέτες στοχεύουν, όπου είναι δυνατόν, στην αποκάλυψη αμερόληπτων σχέσεων μεταξύ εκθέσεων όπως το αλκοόλ ή το κάπνισμα, οι βιολογικοί παράγοντες, το άγχος ή τα χημικά σε θνησιμότητα ή νοσηρότητα. Ο εντοπισμός των αιτιωδών σχέσεων μεταξύ αυτών των εκθέσεων και των αποτελεσμάτων είναι μια σημαντική πτυχή της επιδημιολογίας. Οι σύγχρονοι επιδημιολόγοι χρησιμοποιούν την πληροφορική ως εργαλείο.

Η πειραματική επιδημιολογία περιλαμβάνει τρεις τύπους περιπτώσεων:

- Τυχαιοποιημένες ελεγχόμενες δοκιμές (που χρησιμοποιούνται συχνά για νέα φάρμακα ή δοκιμές φαρμάκων),
- Δοκιμές πεδίου (που διεξάγονται σε άτομα με υψηλό κίνδυνο εμφάνισης ασθένειας)
- Δοκιμές σε κοινότητες (έρευνα για κοινωνικές ασθένειες).

2.2 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ

Η επιδημιολογία διαμορφώθηκε ως επιστήμη τον μεσαίωνα για την μελέτη μεγάλων επιδημιών (χολέρα, ευλογιά, πανώλη).

Ο πρώτος που ασχολήθηκε με τα αντικείμενα της «Επιδημιολογίας» είναι ο Ιπποκράτης (460-357 π.Χ.) στο «Περί αέρος, ύδατος και τόπων». Εκεί κατέγραψε τις

εμπειρικές σχέσεις μεταξύ συγκεκριμένων ασθενειών και του τόπου εκδήλωσης ή εμφάνισης, των συνθηκών διαβίωσης, της διατροφής, κατοικίας, κλίματος και άλλων αιτιών. Όλα αυτά αποτελούν το αντικείμενο της σύγχρονης περιγραφικής επιδημιολογίας.

Πολλά χρόνια αργότερα, ο Graunt (1620-1674) χρησιμοποίησε ποσοτικές μεθόδους για τη μελέτη της θνησιμότητας αγοριών και τις ιδιομορφίες της βρεφικής θνησιμότητας. Ο Farr (1807-1883) έβαλε τις βάσεις για την επιδημιολογική χρήση δημογραφικών δεδομένων εργαζόμενος ως υπεύθυνος Ιατρικών θεμάτων στη στατιστική υπηρεσία της Αγγλίας.

Την ίδια εποχή, ο Snow (1813-1858) απέδειξε ότι η χολέρα προερχόταν από μικρόβιο που μεταδιδόταν από το νερό και την κοπριά. Στο αποτέλεσμα αυτό κατέληξε αφού μελέτησε το νερό από την κεντρική ύδρευση δύο περιοχών ίδιων χαρακτηριστικών (ηλικία, φύλο, κοινωνικοοικονομική κατάσταση): το Lambeth (με καθαρό νερό) και το Soutwork (όπου το νερό περιείχε απόβλητα). Το Lambeth είχε 8 φορές μικρότερη θνησιμότητα από χολέρα. Άρα το νερό ήταν το κλειδί και όχι ο αέρας όπως πίστευαν εκείνη την εποχή.

Στον 20ο αιώνα είχαμε σημαντικές εξελίξεις στην Επιδημιολογία. Μια από τις πιο σημαντικές είναι η σύνδεση του καπνίσματος με τον καρκίνο του πνεύμονα από τον Doll· για λεπτομέρειες βλ. Doll and Peto (1976).

Πολύ σημαντικό γεγονός είναι και η δημιουργία Σχολή δημόσιας Υγείνης το 1922 στο Πανεπιστήμιο του Harvard η οποία περιλαμβάνει τμήματα Επιδημιολογίας και Βιοστατιστικής. Η σχολή αυτή συνέβαλε στον καθορισμό και στην προώθηση της επιστήμης της Επιδημιολογίας και καθιερώθηκε ως ένα από τα καλύτερα Σχολεία στον τομέα αυτό. Για πιο λεπτομερή ανάλυση παραπέμπουμε στα βιβλία του Τριχόπουλου (1982, σελ.4-7) και των McMahon & Trichopoulos (1996).

2.3 ΤΟ ΜΟΝΤΕΛΟ SIR

Το **μοντέλο SIR**⁴ είναι ένα από τα απλούστερα μοντέλα των διαμερισμάτων και πολλά μοντέλα παράγονται από αυτή τη βασική μορφή. Το μοντέλο αποτελείται από **τρία διαμερίσματα**:

- **S** για τον αριθμό ευαίσθητων.
- **I** για τον αριθμό μολυσματικών.
- **R** για τον αριθμό που ανακτήθηκε (ή ανοσοποιητικό).

Αυτό το μοντέλο είναι εύλογα προβλέψιμο για μολυσματικές ασθένειες οι οποίες μεταδίδονται από τον άνθρωπο στον άνθρωπο και όπου η ανάρρωση έχει διαρκή αντίσταση, όπως ιλαρά, παρωτίτιδα και ερυθρά. Αυτές οι μεταβλητές (**S**, **I**, και **R**) αντιπροσωπεύουν τον αριθμό των ατόμων σε κάθε διαμέρισμα σε μια συγκεκριμένη χρονική στιγμή. Για να δείξουμε ότι ο αριθμός των ευπαθών, μολυσμένων και ανακτημένων ατόμων μπορεί να ποικίλει με την πάροδο του χρόνου (ακόμα και αν το συνολικό μέγεθος του πληθυσμού παραμένει σταθερό), κάνουμε τους ακριβείς αριθμούς συνάρτηση του t (time): $S(t)$, $I(t)$, $R(t)$. Για μια συγκεκριμένη ασθένεια σε συγκεκριμένο πληθυσμό, οι λειτουργίες αυτές μπορούν να επεξεργαστούν προκειμένου να προβλεφθούν πιθανές εστίες και να τεθούν υπό έλεγχο.

Ιστορικά, υπάρχουν πάμπολλες επιδημίες κατά τη διάρκεια των ετών. Για παράδειγμα ήδη από τον Θουκυδίδη περιγράφεται η επιδημία πανούκλας στην αρχαία Αθήνα (430 – 428 π.χ.) όταν 1050 από τους 4000 στρατιώτες πεθαίνουν. Ακόμη πιο τραγική είναι η επιδημία πανούκλας στην Ευρώπη το 14ο αιώνα, ή αλλιώς «μαύρος θάνατος» (black death) εκείνη την εποχή, κατά τη διάρκεια της οποίας από ένα πληθυσμό περίπου 85 εκατ. Κατοίκων, πεθαίνει περίπου το 1/3. Πιο σύγχρονη είναι η επιδημία του HIV (Human Immunodeficiency Virus) ο οποίος οδηγεί στο AIDS (Acquired Immune Deficiency Syndrome).

⁴ Compartmental models in epidemiology , https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The_SIR_model , 30 Μαΐου 2008, τελευταία προσπέλαση (μήνας)

Το μοντέλο SIR είναι **δυναμικό**⁵ σε τρεις πτυχές. Όπως υποδηλώνεται από τη μεταβλητή συνάρτηση του t , το μοντέλο είναι δυναμικό δεδομένου ότι οι αριθμοί σε κάθε διαμέρισμα μπορεί να κυμαίνονται διαχρονικά. Η σημασία αυτής της δυναμικής πτυχής είναι πιο εμφανής σε μια ενδημική ασθένεια με σύντομη μολυσματική περίοδο, όπως η ιλαρά. Τέτοιες ασθένειες τείνουν να εμφανίζονται σε κύκλους εμφάνισης εξαιτίας της διακύμανσης του αριθμού των ευπαθών ($S(t)$) με την πάροδο του χρόνου. Κατά τη διάρκεια μιας επιδημίας, ο αριθμός των ευπαθών ατόμων πέφτει γρήγορα, καθώς περισσότεροι από αυτούς μολύνονται και έτσι εισέρχονται στα μολυσματικά και ανακτηθέντα διαμερίσματα. Η ασθένεια δεν μπορεί να ξεσπάσει πάλι μέχρις ότου ο αριθμός των ευπαθών έχει δημιουργηθεί ως αποτέλεσμα της γεννήσεως των απογόνων στο ευπαθές διαμέρισμα. Κάθε μέλος του πληθυσμού εξελίσσεται συνήθως από ευαίσθητα σε μολυσματικά προς ανάκτηση. Αυτό μπορεί να παρουσιαστεί ως ένα διάγραμμα ροής στο οποίο τα πλαίσια αντιπροσωπεύουν τα διαφορετικά διαμερίσματα και τα βέλη τη μετάβαση μεταξύ των διαμερισμάτων, δηλ.



2.4 ΤΟ ΜΟΝΤΕΛΟ SIR ΧΩΡΙΣ ΖΩΤΙΚΗ ΔΥΝΑΜΙΚΗ

Η δυναμική μιας επιδημίας, για παράδειγμα η γρίπη, είναι συχνά πολύ ταχύτερη από τη δυναμική της γέννησης και του θανάτου, επομένως, η γέννηση και ο θάνατος συχνά παραλείπονται σε απλά μοντέλα διαμερισμάτων. Το σύστημα SIR χωρίς τη λεγόμενη ζωτική δυναμική (γέννηση και θάνατος, μερικές φορές ονομάζεται δημογραφία) που περιγράφεται παραπάνω μπορεί να εκφραστεί από το ακόλουθο σύνολο συνήθων διαφορικών εξισώσεων⁶:

⁵ Compartmental models in epidemiology , https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The_SIR_model_is_dynamic_in_three_senses , 30 Μαΐου 2008, τελευταία προσπέλαση (μήνας)

⁶ Hethcote H (2000). "[The Mathematics of Infectious Diseases](#)". *SIAM Review*. **42**(4): 599–653.

— — — — —

Όπου:

- **(S)** είναι το απόθεμα ευπαθούς πληθυσμού
- **(I)** είναι το απόθεμα μολυσμένων
- **(R)** είναι το απόθεμα του ανακτηθέντος πληθυσμού.

Το μοντέλο αυτό προτάθηκε για πρώτη φορά από τους O. Kermack και Anderson Gray McKendrick ως μια ειδική περίπτωση την οποία πλέον αποκαλούμε θεωρία Kermack-McKendrick.

Το σύστημα αυτό είναι μη γραμμικό και δεν δέχεται μια γενική αναλυτική λύση. Παρ'όλα αυτά, σημαντικά αποτελέσματα μπορούν να προκύψουν και με τις μεθόδους Monte Carlo, όπως ο αλγόριθμος Gillespie.

Καταρχάς σημειώστε ότι από:

— — —

συνεπάγεται ότι:

εκφράζοντας με μαθηματικούς όρους τη σταθερότητα του πληθυσμού N . Αξίζει να σημειωθεί ότι η παραπάνω σχέση υποδηλώνει ότι πρέπει να μελετήσουμε μόνο την εξίσωση για δύο από τις τρεις μεταβλητές. Δεύτερον, σημειώνουμε ότι η δυναμική της μολυσματικής τάξης εξαρτάται από τον ακόλουθο λόγο:

—

Αυτός ο λόγος, είναι ο λεγόμενος βασικός αριθμός αναπαραγωγής (που ονομάζεται επίσης βασικός λόγος αναπαραγωγής). Αυτή η αναλογία προκύπτει ως ο αναμενόμενος αριθμός νέων μολύνσεων (αυτές οι νέες μολύνσεις ονομάζονται μερικές φορές

δευτερογενείς λοιμώξεις) από μια μόνη μόλυνση σε έναν πληθυσμό όπου όλα τα άτομα είναι ευαίσθητα^{7,8}. Διαιρώντας την πρώτη διαφορική εξίσωση με την τρίτη, διαχωρίζοντας τις μεταβλητές και ολοκληρώνοντας παίρνουμε:

Όπου $S(0)$ και $R(0)$ είναι οι αρχικοί αριθμοί, αντίστοιχα, ευαίσθητων και απομακρυσμένων ατόμων. Έτσι, στο όριο $t \rightarrow \infty$, το ποσοστό των ανακτηθέντων ατόμων υπακούει στην ακόλουθη υπερβατική εξίσωση:

Αυτή η εξίσωση μας δείχνει ότι στο τέλος μιας επιδημίας, εκτός αν $R(0) > 1$, δεν έχουν ανακάμψει όλα τα άτομα του πληθυσμού, έτσι ορισμένα πρέπει να παραμείνουν ευαίσθητα. Αυτό σημαίνει ότι το τέλος μιας επιδημίας προκαλείται από τη μείωση του αριθμού των μολυσμένων ατόμων και όχι από την απόλυτη έλλειψη ευαίσθητων ατόμων. Ο ρόλος του βασικού αριθμού αναπαραγωγής είναι εξαιρετικά σημαντικός. Στην πραγματικότητα, με την επανεγγραφή της εξίσωσης για τα μολυσματικά άτομα, η εξίσωση έχει ως εξής:

το οποίο σημαίνει ότι εάν $R(0) < 1$ τότε $R(t) \rightarrow 0$ θα υπάρξει μια επιδημική έξαρση με αύξηση του αριθμού των μολυσματικών (που μπορεί να φτάσει σε ένα σημαντικό κλάσμα του πληθυσμού). Αντίθετα, εάν $R(0) > 1$ τότε $R(t) \rightarrow R(0)$, τότε ανεξάρτητα από το αρχικό μέγεθος του ευαίσθητου πληθυσμού, η ασθένεια δεν μπορεί ποτέ να προκαλέσει την κατάλληλη επιδημική έκρηξη. Κατά συνέπεια, είναι σαφές ότι ο βασικός αριθμός αναπαραγωγής είναι εξαιρετικά σημαντικός.

⁷ Bailey, Norman T. J. (1975). *The mathematical theory of infectious diseases and its applications* (2nd ed.). London: Griffin. ISBN 0-85264-231-8.

⁸ Sonia Altizer; Nunn, Charles (2006). *Infectious diseases in primates: behavior, ecology and evolution*. Oxford Series in Ecology and Evolution. Oxford [Oxfordshire]: Oxford University Press. ISBN 0-19-856585-2.

2.5 ΤΟ ΜΟΝΤΕΛΟ SIR ΜΕ ΖΩΤΙΚΗ ΔΥΝΑΜΙΚΗ

Δεδομένου ότι ένας πληθυσμός χαρακτηρίζεται από ένα ποσοστό θνησιμότητας και ένα ποσοστό γεννήσεων και όπου μεταδίδεται μια μεταδοτική ασθένεια, το μοντέλο⁹ με μετάδοση μαζικής ενέργειας είναι:

—

—

—

για την οποία είναι η **ισορροπία χωρίς ασθένεια** (Disease Free Equilibrium “DFE”) όπου:

—

τότε μπορούμε να υπολογίσουμε έναν βασικό αριθμό αναπαραγωγής:

—

το οποίο έχει ιδιότητες κατωφλίου. Στην πραγματικότητα, ανεξάρτητα από τις βιολογικά σημαντικές αρχικές τιμές, μπορεί κανείς να δείξει ότι:

—

— — —

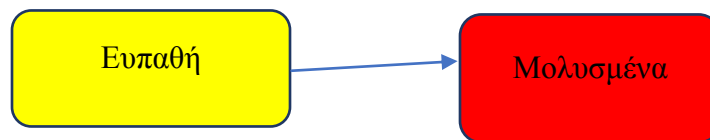
⁹ The SIR model ,

https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The_SIR_model , 30 Μαΐου 2008, τελευταία προσπέλαση (μήνας)

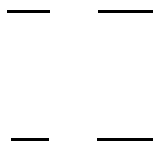
Το σημείο αυτό ονομάζεται **ενδημική ισορροπία** (Endemic Equilibrium EE)

2.6 ΤΟ ΜΟΝΤΕΛΟ SIS

Ορισμένες λοιμώξεις, για παράδειγμα αυτές που προέρχονται από το κοινό κρυολόγημα και τη γρίπη, δεν παρέχουν μακροχρόνια ανοσία. Τέτοιες λοιμώξεις δεν δίνουν ανοσοποίηση μετά την ανάκτηση από τη μόλυνση, και τα άτομα γίνονται επιρρεπή ξανά.

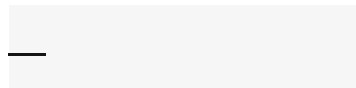


Έχουμε το μοντέλο¹⁰:

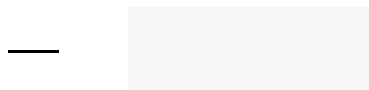


Σημειώστε ότι με N τον συνολικό πληθυσμό: — —

συνεπάγεται ότι:



δηλαδή η δυναμική των μολυσματικών κυριαρχείται από μια λογική εξίσωση, έτσι ώστε:



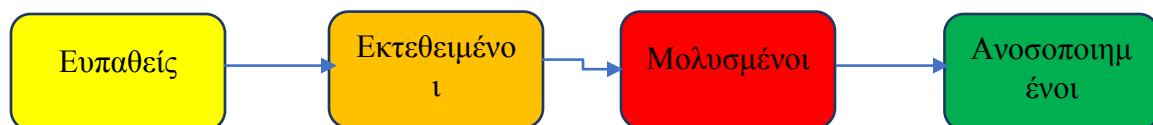
¹⁰ The SIS model, https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The_SIS_model, 30 Μαΐου 2008, τελευταία προσπέλαση (μήνας)

Είναι δυνατόν να βρεθεί μια αναλυτική λύση σε αυτό το μοντέλο (κάνοντας μετασχηματισμό μεταβλητών: και την υποκατάσταση αυτού στις εξισώσεις μέσου πεδίου), ¹¹ έτσι ώστε ο βασικός ρυθμός αναπαραγωγής να είναι μεγαλύτερος από την ενότητα. Άρα έχουμε:

όπου — είναι ο ενδημικός μολυσμένος πληθυσμός,
και —. Όταν το σύστημα θεωρείται κλειστό, τότε ο ευπαθής πληθυσμός είναι:

2.7 TO MONTELO SEIR

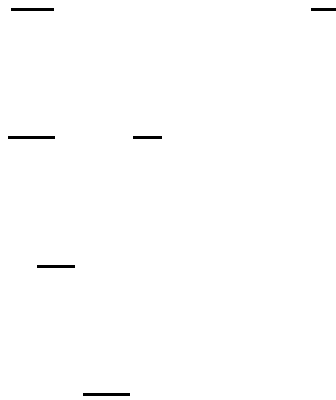
Παραλλαγές του μοντέλου **SIR**, με περισσότερες καταστάσεις, αποτελούν τα μοντέλα **SEIS** και **SEIR**. Στο μοντέλο **SEIS**, η επιπρόσθετη κλάση **E** (exposed), αφορά τα άτομα τα οποία έχουν εκτεθεί στον ιό, έχουν ασθενήσει αλλά για κάποιο χρονικό διάστημα δεν μπορούν να μεταδώσουν την ασθένεια. Μετά το πέρας αυτού του διαστήματος, τα άτομα μεταβαίνουν στην κατάσταση **I** (infected) όπου είναι ασθενείς και μπορούν να μολύνουν κάποιον. Επιπλέον, δεν τίθεται θέμα ανοσίας σε αυτό το μοντέλο καθώς η μετάβαση από την κατάσταση **I**, είναι μόνο προς την κατάσταση **S** (susceptible).



Υποθέτοντας ότι η περίοδος επώασης είναι μια τυχαία μεταβλητή με εκθετική κατανομή με την παράμετρο a (δηλαδή η μέση περίοδος επώασης είναι $\frac{1}{a}$), και επίσης

¹¹ Hethcote, H. W. (1989). in *Three Basic Epidemiological Models, Applied Mathematical Ecology, Vol. 18*. Berlin: Springer.

υποθέτοντας την ύπαρξη ζωτικής δυναμικής με το ποσοστό γεννήσεως που ισούται με το ποσοστό θνησιμότητας, έχουμε το μοντέλο¹²:



Έχουμε β , αλλά αυτό είναι μια σταθερά, λόγω της (εκφυλισμένης) παραδοχής ότι τα ποσοστά γεννήσεων και θανάτου είναι ίσα. Γενικά το N είναι μια μεταβλητή.

Για αυτό το μοντέλο, ο βασικός αριθμός αναπαραγωγής είναι:

$$\frac{\beta}{\gamma}$$

Όπως στο μοντέλο SIR, έτσι και σε αυτή την περίπτωση έχουμε μια ισορροπία χωρίς νόσους $S = N$ και μια ενδημική ισορροπία $E = I = \frac{\beta N}{\gamma}$ και μπορούμε να δείξουμε ότι ανεξάρτητα από βιολογικά σημαντικές αρχικές συνθήκες έχουμε:

ισχύει ότι:

¹² The SEIR model , https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The_SEIR_model , 30 Μαΐου 2008 , τελευταία προσπέλαση (μήνας)

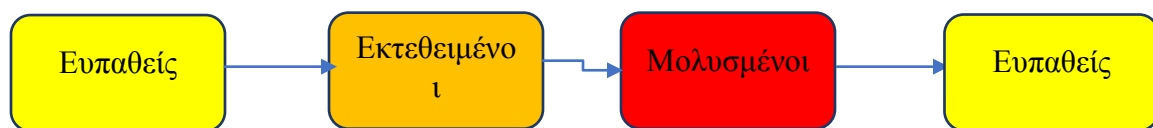
Σε περίπτωση περιοδικά μεταβαλλόμενης ταχύτητας επαφής η συνθήκη για την παγκόσμια ελκυστικότητα του DFE είναι ότι το ακόλουθο γραμμικό σύστημα με περιοδικούς συντελεστές:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

είναι σταθερό.

2.8 TO MONTEΛΟ SEIS

Το μοντέλο **SEIS**¹³ λαμβάνει υπόψη την εκτεθειμένη ή λανθάνουσα περίοδο της νόσου, δίνοντας ένα επιπλέον διαμέρισμα,



Σε αυτό το μοντέλο μια λοίμωξη δεν αφήνει καμία ασυλία έτσι τα άτομα που έχουν ανακάμψει επιστρέφουν στο να είναι ευπαθή και πάλι, και έτσι κινούνται πίσω στο διαμέρισμα. Οι παρακάτω διαφορικές εξισώσεις περιγράφουν αυτό το μοντέλο:

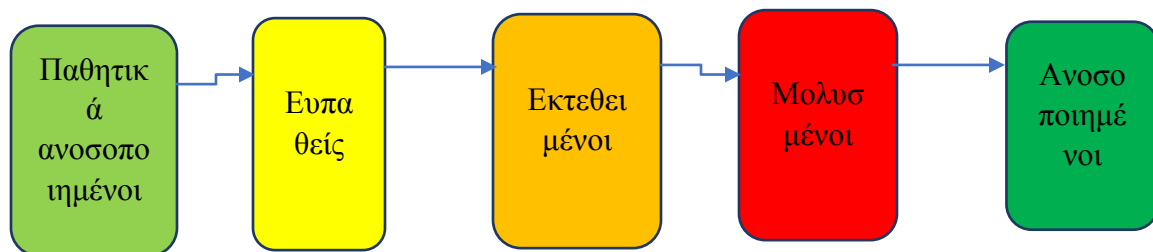
—

¹³ The SEIS model,

https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The_SEIS_model, 30 Μαΐου 2008, τελευταία προσπέλαση (μήνας)

2.9 ΤΟ ΜΟΝΤΕΛΟ MSEIR

Στην περίπτωση μιας ασθένειας, με τους παράγοντες της παθητικής ανοσίας και μιας περιόδου λανθάνουσας κατάστασης, υπάρχει ακόμα μια παραλλαγή του μοντέλου **SIR**, το μοντέλο **MSEIR**¹⁴.



Πιο αναλυτικά, το μοντέλο MSEIR ορίζεται από τις παρακάτω εξισώσεις:

¹⁴ The SIR model , https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The_MSEIR_model , 30 Μαΐου 2008 , τελευταία πρόσβαση 27/9/2019.

2.10 ΚΑΤΩΤΑΤΑ ΟΡΙΑ ΓΙΑ ΑΥΘΑΙΡΕΤΑ ΜΟΝΤΕΛΑ ΚΑΤΑΡΡΑΚΤΗ ΣΕ ΑΥΘΑΙΡΕΤΑ ΔΙΚΤΥΑ

Σε αυτή την υποενότητα, οι συγγραφείς δίνουν απάντηση στο εξής ερώτημα: Ποιες συνθήκες πρέπει να ισχύουν έτσι ώστε η εξάπλωση ενός ιού να οδηγήσει σε επιδημία; Ποιο είναι το αντίστοιχο κατώφλι; Συγκεκριμένα, δίνουν απάντηση στο ερώτημα, χρησιμοποιώντας ένα γενικευμένο μοντέλο διακριτού χρόνου που υπερκαλύπτει τα γνωστά επιδημιολογικά μοντέλα και αποδεικνύουν ότι η απάντηση έγκειται στη μέγιστη ιδιοτιμή του μητρώου γειτνίασης του γραφήματος. Σε αυτή την υποενότητα, περιγράφουμε αναλυτικά την προσέγγιση που ακολούθησαν. Η συγκεκριμένη εργασία, αποτελεί την πρώτη προσπάθεια για ενοποίηση των παρατηρήσεων που κατά καιρούς έχουν προκύψει από την εξάπλωση ιών σε διαφορετικές τοπολογίες. Το βασικό αποτέλεσμα που παρουσιάζεται, είναι ότι ανεξαρτήτως επιδημιολογικού μοντέλου, η συνεισφορά του δικτύου στην εξάπλωση του ιού, αντιπροσωπεύεται από μία παράμετρο: τη μέγιστη ιδιοτιμή του μητρώου γειτνίασης που αναπαριστά το δίκτυο. Το μοντέλο που χρησιμοποιήθηκε στη συγκεκριμένη περίπτωση, αποτελεί ένα γενικευμένο μοντέλο που υπερκαλύπτει την πλειοψηφία των επιδημιολογικών μοντέλων που περιγράψαμε στην προηγούμενη ενότητα και οι συγγραφείς το ονομάζουν . Όπως αναφέραμε και σε προηγούμενη ενότητα, τα επιδημιολογικά μοντέλα, περιλαμβάνουν καταστάσεις που περιλαμβάνονται σε μία από τις παρακάτω κλάσεις:

- **Susceptible $S(t)$** : Οι κόμβοι σε μία κατάσταση αυτής της κλάσης, μπορούν να μολυνθούν από οποιονδήποτε γειτονικό κόμβο που είναι μολυσματικός.
- **Infected** : Οι κόμβοι σε αυτή την κλάση, θεωρούνται μολυσματικοί με την έννοια ότι μπορούν να μεταφέρουν τη μόλυνση σε κάποιον γειτονικό κόμβο με μία πιθανότητα μετάδοσης. Φυσικά, η πιθανότητα μετάδοσης μπορεί να είναι 0, το οποίο συνεπάγεται ότι ο κόμβος έχει εκτεθεί στον ιό και μπορεί να το μεταφέρει αλλά δεν έχει ο ίδιος μολυνθεί.
- **Vigilant/vaccinated** : Οι κόμβοι που βρίσκονται σε κατάσταση αυτής της κλάσης (πχ κατάσταση R στο μοντέλο SIR), δε μπορούν να μολυνθούν ούτε να

μεταδώσουν κάποιον ιό. Χρησιμοποιώντας καταστάσεις από τις κλάσεις που περιγράψαμε προηγουμένως, το γενικευμένο μοντέλο περιλαμβάνει αυθαίρετο αριθμό από καταστάσεις της κλάσης Susceptible (S), 2 καταστάσεις από την κλάση Infected (I2) και αυθαίρετο αριθμό από καταστάσεις της κλάσης Vigilant (V). Οι συγγραφείς, έχουν κάνει δύο βασικές υποθέσεις:

- **Μόλυνση από γειτονικούς κόμβους:** Ο μόνος δυνατός τρόπος να μολυνθεί κάποιος κόμβος, είναι μέσω κάποιου γείτονα. Δεν υπάρχει κάποια ενδογενής μετάβαση από την κλάση Susceptible στην κλάση Infected παρά μόνο λόγω εξωγενών παραγόντων.
- **Αρχική κατάσταση στην κλάση Infected :** Για το μικρό αριθμό μοντέλων που διαθέτουν περισσότερες από μία κατάσταση στην κλάση Infected, ισχύει ότι οποιαδήποτε εξωγενής μετάβαση, μεταφράζεται σε μετάβαση από την κλάση Susceptible στην κλάση Infected.

Λαμβάνοντας υπ' όψιν τις προηγούμενες υποθέσεις, οι συγγραφείς προσέγγισαν το γενικευμένο μοντέλο, χρησιμοποιώντας ένα μη γραμμικό δυναμικό σύστημα διακριτού χρόνου που μοντελοποιεί την μετάβαση των κόμβων ανάμεσα στις καταστάσεις όπου η κατάσταση των κόμβων κάθε χρονική στιγμή, αναπαρίσταται σαν ένα διάνυσμα . Για παράδειγμα, για ένα μοντέλο m καταστάσεων σε ένα δίκτυο N κόμβων, το διάνυσμα

θα είναι:

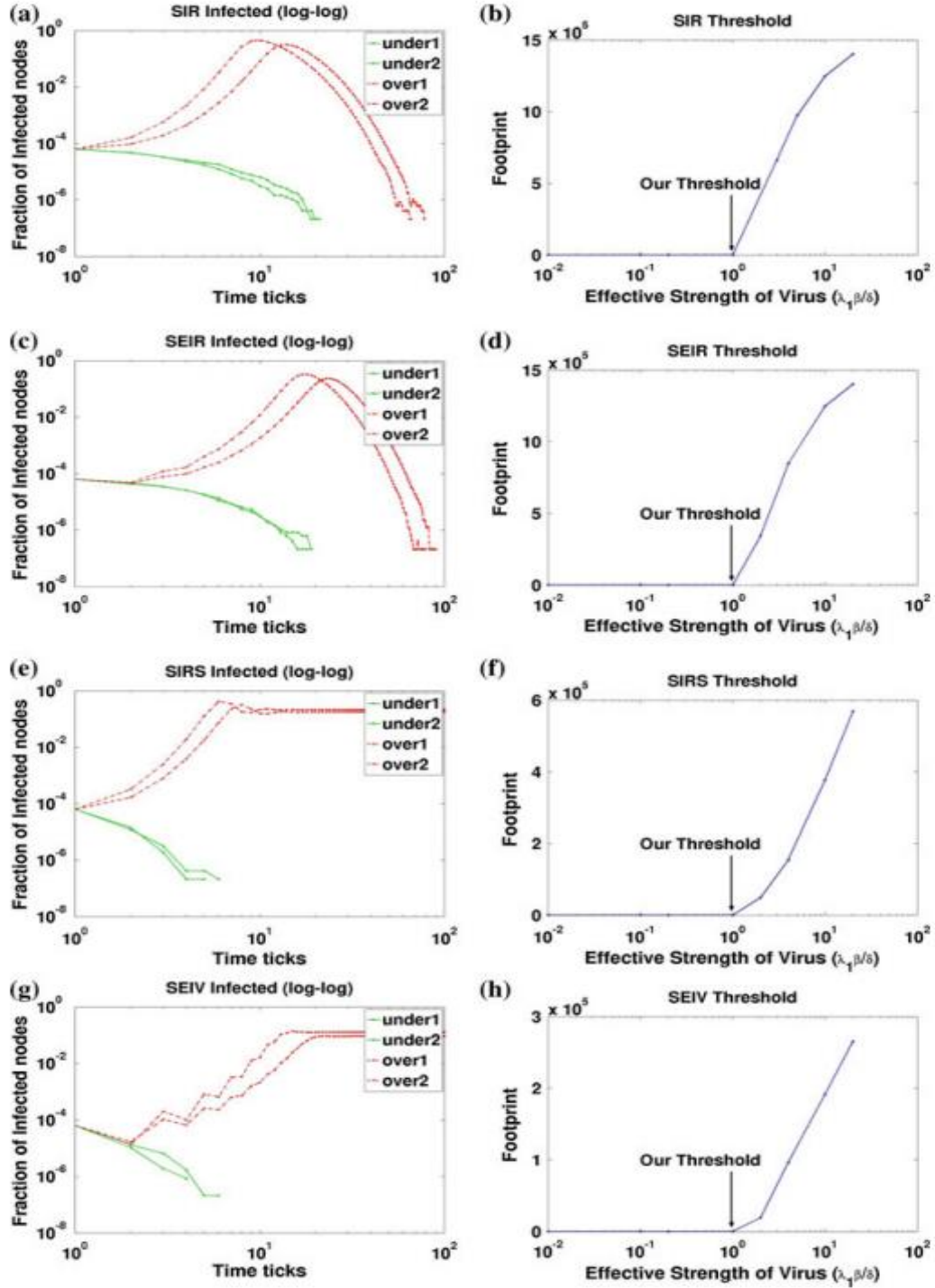
T όπου είναι η

πιθανότητα ο κόμβος j να είναι στην κατάσταση i τη χρονική στιγμή t . Αντίστοιχα το μη γραμμικό δυναμικό σύστημα που μοντελοποιεί τις μεταβολές στις τιμές του διανύσματος

είναι της μορφής όπου g είναι μία μη γραμμική συνάρτηση που χρησιμοποιεί ως όρισμα το διάνυσμα . Το βασικό θεώρημα το οποίο απέδειξαν, περιγράφεται παρακάτω:

Για επιδημιολογικά μοντέλα που ικανοποιούν τις αρχικές υποθέσεις και για οποιοδήποτε αυθαίρετο και μη-κατευθυνόμενο γράφημα με μητρώο γειτνίασης A και μέγιστη ιδιοτιμή λ_1 , η ικανή συνθήκη για να είναι το σύστημα ευσταθές, είναι: $s < 1$ όπου s είναι η δύναμη και είναι ίση: και είναι μία σταθερά που εξαρτάται από το επιδημιολογικό μοντέλο που χρησιμοποιείται σε κάθε περίπτωση.

Το κρίσιμο σημείο είναι στην περίπτωση που ισχύει $R_0 < 1$. Αυτό το σημείο αποτελεί το λεγόμενο επιδημιολογικό κατώφλι (epidemic threshold). Όταν η δύναμη του ιού είναι μικρότερη από 1, ο ιός εξουδετερώνεται ταχύτατα σε σύγκριση με την περίπτωση που η δύναμή του βρίσκεται άνω του κατωφλίου (ανάλογα πάντα με το επιδημιολογικό μοντέλο). Τα αποτελέσματα που προκύπτουν για τα επιμέρους μοντέλα, συνοψίζονται στην παρακάτω εικόνα.



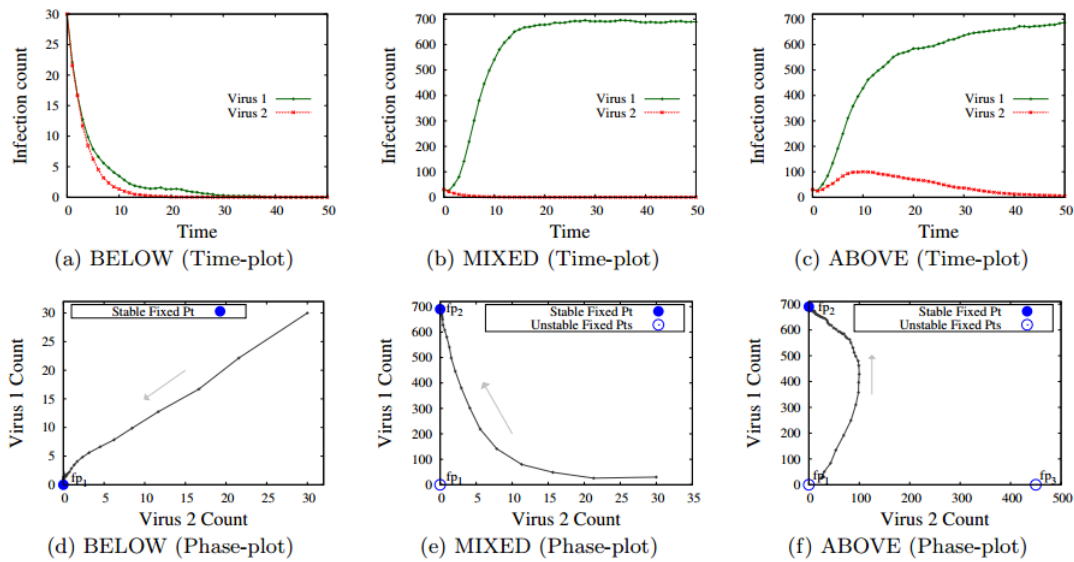
Εικόνα 1. Πειραματικά αποτελέσματα για τα επιμέρους μοντέλα του $S * I 2V$ * ¹⁵

¹⁵ B. A. Prakash, D. Chakrabarti, M. Faloutsos, N. Valler, and C. Faloutsos. Threshold conditions for arbitrary cascade models on arbitrary networks. In Proceedings of the 2011 IEEE 11th International Conference on Data Mining, ICDM '11, pages 537–546, Washington, DC, USA, 2011. IEEE Computer Society.

2.11 ΜΟΝΤΕΛΟ ΣΥΝΕΧΟΥΣ ΧΡΟΝΟΥ WINNER TAKES ALL

Στη συγκεκριμένη εργασία, οι συγγραφείς προσπαθούν να μοντελοποιήσουν το φαινόμενο δύο ανταγωνιστικών προϊόντων/ιών που εξαπλώνονται στο ίδιο δίκτυο. Το ερώτημα που προκύπτει είναι ποιος από τους δύο θα «νικήσει» (θα καταλάβει το μεγαλύτερο μέρος του δικτύου). Οι συγγραφείς απαντούν στο ερώτημα αποδεικνύοντας ότι υπό ρεαλιστικές συνθήκες ο δυνατότερος ιός από τους δύο, εξουδετερώνει τον άλλο και καταλαμβάνει όλο το δίκτυο, ανεξάρτητα από την τοπολογία του δικτύου. Το μοντέλο που χρησιμοποίησαν είναι το SIS που θυμίζει το γνωστό επιδημιολογικό μοντέλο SIS με την διαφοροποίηση ότι οι κόμβοι όταν μολυνθούν από τον ιό 1, μεταβαίνουν στην κατάσταση S ενώ αντίστοιχα όταν μολυνθούν από τον ιό 2, μεταβαίνουν στην κατάσταση I . Οι συγγραφείς, έχουν κάνει τρεις βασικές υποθέσεις:

- **Αμοιβαία ανοσία** (mutual immunity): Ένας κόμβος που έχει μολυνθεί από τον ένα ιό, δεν μπορεί να μολυνθεί από τον άλλο. Δηλαδή, ένας κόμβος δεν μπορεί να είναι την ίδια χρονική περίοδο «ασθενής» και από τους 2 ιούς.
- **Συνεκτικότητα**: Το γράφημα είναι συνεκτικό με την έννοια ότι υπάρχει μονοπάτι από κάθε κόμβο προς όλους τους υπόλοιπους κόμβους του δικτύου.
- **Fair-play**: Το δίκτυο είναι fair-play με την έννοια ότι όλοι οι κόμβοι έχουν την ίδια συμπεριφορά απέναντι στους δύο ανταγωνιστικούς ιούς. Αυτό πρακτικά συνεπάγεται ότι όλοι οι κόμβοι έχουν την ίδια πιθανότητα να μολυνθούν από τον ιό 1 και αντίστοιχα, έχουν την ίδια πιθανότητα να μολυνθούν από τον ιό 2 και το ίδιο ισχύει για τις πιθανότητες ίασης. Με βάση τις προηγούμενες υποθέσεις, κατασκεύασαν ένα δυναμικό σύστημα που μοντελοποιεί τη διαδικασία μόλυνσης του δικτύου και παρουσίασαν τις συνθήκες που πρέπει να ικανοποιούνται ώστε το σύστημα να είναι ευσταθές στην περίπτωση της κλίμακας, του γραφήματος barbell αλλά και αυθαίρετων γραφημάτων.



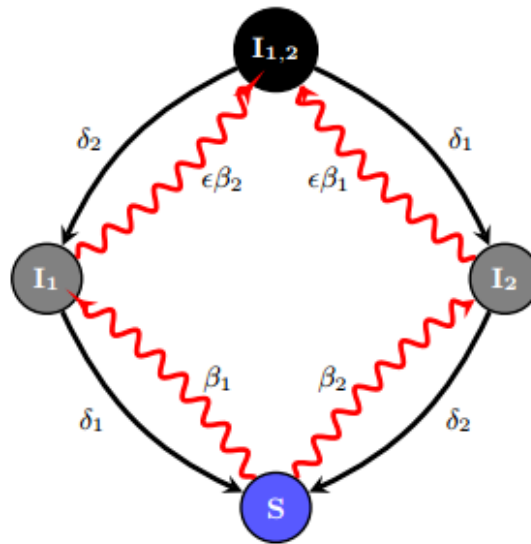
Εικόνα 2. Πειράματα στο δίκτυο AS-OREGON για τα 3 σημεία ισορροπίας (αριθμός μολυσμένων κόμβων vs χρόνος, αριθμός μολυσμένων κόμβων ιού 1 vs αριθμός μολυσμένων κόμβων ιού 2) ¹⁶

Επιπλέον, απέδειξαν ότι τα διαθέσιμα σημεία ισορροπίας, ανεξαρτήτως τοπολογίας, είναι 3:

- **BELOW**: Οι ιοί έχουν δύναμη χαμηλότερη από το κατώφλι,
 - **MIXED**: Ο ένας ιός έχει δύναμη άνω του κατωφλίου και ο άλλος χαμηλότερη από το κατώφλι,
 - **ABOVE**: Οι ιοί έχουν δύναμη άνω του κατωφλίου,
- Φυσικά, για κάθε διαφορετικό σενάριο, ένα από τα διαθέσιμα σημεία ισορροπίας θα είναι σταθερό. Χρησιμοποιώντας, τα σημεία αυτά, κατέληξαν στο παρακάτω συμπέρασμα: Δοθέντος ενός αυθαίρετου συνεκτικού γραφήματος με μητρώο γειτνίασης και τις παραμέτρους του μοντέλου, τότε ο ιός 1 θα κυριαρχήσει στο δίκτυο και ο ιός 2 θα εξουδετερωθεί στο σημείο ισορροπίας, αν ο ιός 1 έχει δύναμη άνω του κατωφλίου () και η δύναμη του ιού 1 είναι μεγαλύτερη από τη δύναμη του ιού 2 (). Τέλος, για καθένα από τα

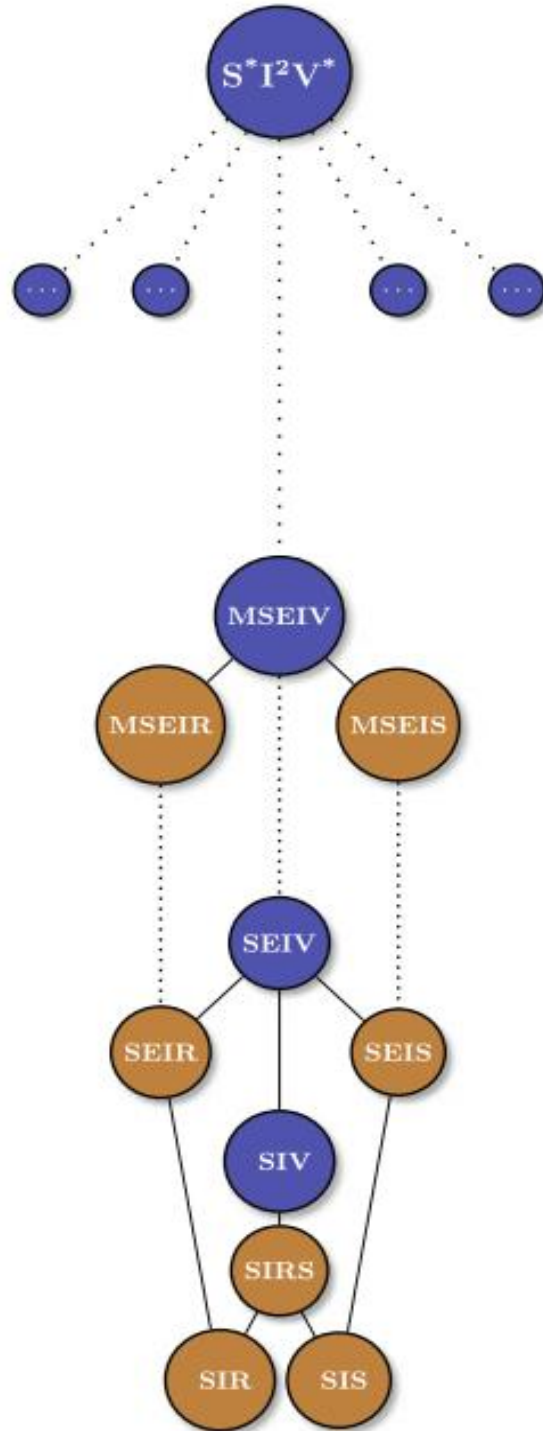
¹⁶ B. A. Prakash, A. Beutel, R. Rosenfeld, and C. Faloutsos. Winner takes all: Competing viruses or ideas on fair-play networks. In Proceedings of the 21st International Conference on World Wide Web, WWW '12, pages 1037–1046, New York, NY, USA, 2012. ACM.

σημεία ισορροπίας, πραγματοποίησαν πειράματα με συνθετικά αλλά και πραγματικά δεδομένα, επαληθεύοντας το αποτέλεσμα στο οποίο κατέληξαν, ο δυνατότερος ιός (με δύναμη μεγαλύτερη από το κατώφλι), κυριαρχεί στο δίκτυο (winner takes all).



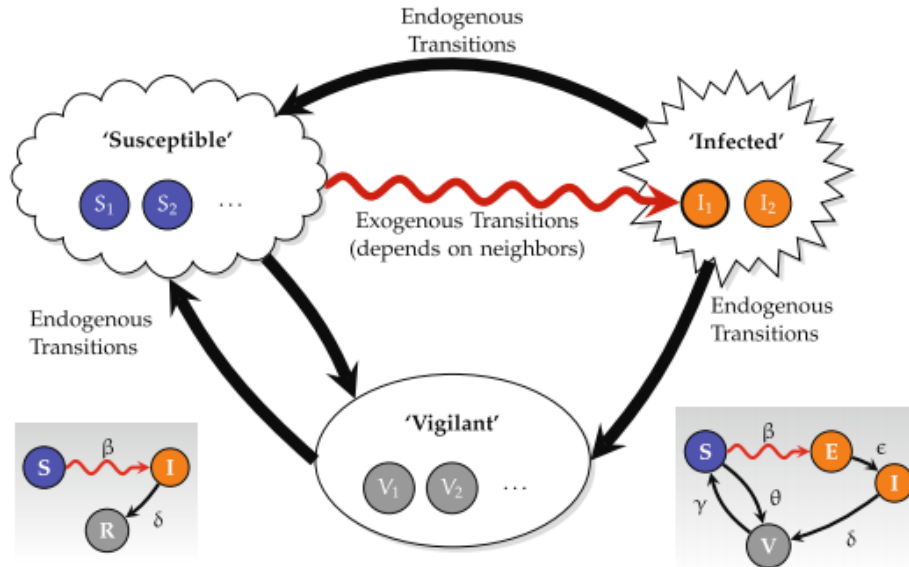
Εικόνα 3. Διάγραμμα καταστάσεων ενός κόμβου στο μοντέλο μερικού ανταγωνισμού S ¹⁷

¹⁷ A. Beutel, B. A. Prakash, R. Rosenfeld, and C. Faloutsos. Interacting viruses in networks: Can both survive? In Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, pages 426–434, New York, NY, USA, 2012. ACM.



Εικόνα 4. Γενικευμένο επιδημιολογικό μοντέλο $S * I^2 V *$ ¹⁸

¹⁸ B. A. Prakash, D. Chakrabarti, M. Faloutsos, N. Valler, and C. Faloutsos. Threshold conditions for arbitrary cascade models on arbitrary networks. In Proceedings of the 2011 IEEE 11th International Conference on Data Mining, ICDM '11, pages 537–546, Washington, DC, USA, 2011. IEEE Computer Society.



Εικόνα 5. Διάγραμμα καταστάσεων & μεταβάσεις στο επιδημιολογικό μοντέλο SIV ¹⁹

Από το παραπάνω διάγραμμα παρατηρούμε ότι ο αριθμός των ευπαθών ατόμων αρχικά μειώνεται σημαντικά και έπειτα σταθεροποιείται ενώ παράλληλα ο αριθμός των μολυσμένων αυξάνεται το ίδιο χρονικό διάστημα το οποίο δηλώνει και την εξάπλωση της επιδημίας. Ο αριθμός των ανακτημένων αυξάνεται με μικρό ρυθμό και τελικά σταθεροποιείται, σε αντίθεση όταν $R_0 < 1$, όπου η αύξηση γίνεται με μεγαλύτερο ρυθμό.

¹⁹ B. A. Prakash, D. Chakrabarti, M. Faloutsos, N. Valler, and C. Faloutsos. Threshold conditions for arbitrary cascade models on arbitrary networks. In Proceedings of the 2011 IEEE 11th International Conference on Data Mining, ICDM '11, pages 537–546, Washington, DC, USA, 2011. IEEE Computer Society.

ΚΕΦΑΛΑΙΟ 3: ΕΙΣΑΓΩΓΗ ΣΤΗΝ PYTHON

3.1 ΕΙΣΑΓΩΓΗ

Η Python είναι διερμηνευόμενη (interpreted), γενικού σκοπού (general-purpose) και υψηλού επιπέδου, γλώσσα προγραμματισμού. Ανήκει στις γλώσσες προστακτικού προγραμματισμού (Imperative programming) και υποστηρίζει τόσο τον διαδικαστικό (procedural programming) όσο και τον αντικειμενοστραφές (object-oriented programming) προγραμματισμό. Είναι δυναμική γλώσσα προγραμματισμού (dynamically typed) και υποστηρίζει συλλογή απορριμμάτων (garbage collection ή GC).

Δημιουργήθηκε από τον Ολλανδό Γκίντο βαν Ρόσσουμ το 1990 και κυκλοφόρησε για πρώτη φορά το 1991. Ο κύριος στόχος της είναι η αναγνωσιμότητα του κώδικά της και η ευκολία χρήσης της. Το συντακτικό της επιτρέπει στους προγραμματιστές να εκφράσουν έννοιες σε λιγότερες γραμμές κώδικα από ότι θα ήταν δυνατόν σε γλώσσες όπως η C++ ή η Java. Διακρίνεται λόγω του ότι έχει πολλές βιβλιοθήκες που διευκολύνουν ιδιαίτερα αρκετές συνηθισμένες εργασίες και για την ταχύτητα εκμάθησής της. Οι διερμηνευτές της Python είναι διαθέσιμοι για εγκατάσταση σε πολλά λειτουργικά συστήματα, επιτρέποντας στην Python την εκτέλεση κώδικα σε ευρεία γκάμα συστημάτων. Χρησιμοποιώντας εργαλεία τρίτων, όπως το Py2exe ή το Pyinstaller, ο κώδικας της Python μπορεί να πακεταριστεί σε αυτόνομα εκτελέσιμα προγράμματα για μερικά από τα πιο δημοφιλή λειτουργικά συστήματα, επιτρέποντας τη διανομή του βασισμένου σε Python λογισμικού για χρήση σε αυτά τα περιβάλλοντα χωρίς να απαιτείται εγκατάσταση του διερμηνευτή της Python. Η Python αναπτύσσεται ως ανοιχτό λογισμικό (open source) και η διαχείρισή της γίνεται από τον μη κερδοσκοπικό οργανισμό Python Software Foundation. Ο κώδικας διανέμεται με την άδεια Python Software Foundation License, η οποία είναι συμβατή με την GPL. Το όνομα της γλώσσας προέρχεται από την ομάδα των Άγγλων κωμικών Μόντυ Πάιθον και δεν έχει καμιά σχέση με το φίδι πύθωνα, παρότι το λογότυπό της παραπέμπει σε κάτι τέτοιο.

3.2 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ

Όπως αναφέρει ο Λεβεντέας Δ. (2008) στο σύγγραμμά του, η Python ήταν γλώσσα σεναρίων που χρησιμοποιούνταν στο λειτουργικό σύστημα Amoeba, ικανή και για κλήσεις συστήματος. Αναφέρει, επίσης, ότι η Python 2.0 κυκλοφόρησε στις 16 Οκτωβρίου του 2000, ενώ στις 3 Δεκεμβρίου 2008 κυκλοφόρησε η έκδοση 3.0 (γνωστή και ως py3k ή python 3000). Πολλά από τα καινούργια χαρακτηριστικά αυτής της έκδοσης έχουν μεταφερθεί στις εκδόσεις 2.6 και 2.7 που είναι προς τα πίσω συμβατές. Η Python 3.0 είναι ιστορικά η πρώτη γλώσσα προγραμματισμού που σπάει την προς τα πίσω συμβατότητα με προηγούμενες εκδόσεις ώστε να διορθωθούν κάποια λάθη που υπήρχαν σε προγενέστερες εκδόσεις και να καταστεί ακόμα πιο σαφής ο απλός τρόπος με τον οποίο μπορούν να γίνουν κάποια πράγματα.

Όσο για τον Γκίντο βαν Ρόσσουμ, αναφέρεται ότι πρωτοστατούσε στην κοινότητα της python μέχρι και τον Ιούλιο του 2018, οπότε και παραιτήθηκε. Η δημιουργία της Python ξεκίνησε το 1989 από τον Ολλανδό Γκίντο βαν Ρόσσουμ στο ερευνητικό κέντρο Centrum Wiskunde & Informatica (CWI), το οποίο επιτελεί βασική έρευνα στο πεδίο των Μαθηματικών και της Επιστήμης Υπολογιστών.

Η Python θεωρείται διάδοχος της γλώσσας προγραμματισμού ABC, μιας και αυτή υπήρξε η βασική πηγή έμπνευσης για τη δημιουργία της. Αρχικά, χρησιμοποιήθηκε ως γλώσσα σεναρίων για το κατανεμημένο λειτουργικό σύστημα Amoeba. Όπως αναφέρει ο ίδιος ο Γκίντο βαν Ρόσσουμ, η πιο καινοτόμα συνεισφορά του στην επιτυχία της Python ήταν το ότι αυτή είχε τη δυνατότητα να επεκτείνεται εύκολα. Το μοντέλο της Python σχεδιάστηκε με τέτοιο τρόπο ώστε να είναι ευέλικτα επεκτάσιμο, να παρέχει ενσωματωμένα στοιχεία (εντολές, τύπους αντικειμένων, κ.λπ.), αλλά και να δίνει τη δυνατότητα στους προγραμματιστές να προσθέτουν τα δικά τους στοιχεία ανάλογα με τις ανάγκες τους και το σύστημα που χρησιμοποιούν. Όπως προαναφέρθηκε, η πρώτη έκδοση κυκλοφόρησε το 1991. Η ονομασία της οφείλεται στη δημοφιλή κωμική σειρά Monty Python's Flying Circus του BBC της Μεγάλης Βρετανίας (δεκαετία του '70). Ο Γκίντο βαν Ρόσσουμ ήθελε ένα όνομα για τη νέα γλώσσα προγραμματισμού, το οποίο να είναι μικρό, μοναδικό και ελαφρώς μυστήριο. Μάλιστα ενθαρρύνεται η χρήση στην

τεκμηρίωση της Python αναφορών στη σάτιρα της παραπάνω κωμικής σειράς. Δεν υπάρχει συνεπώς καμία συσχέτιση με ερπετά και το γνωστό φίδι πύθωνα.

Σύμφωνα με τον Λεβεντέα Δ. (2008), η Python εξελίσσεται γρήγορα μέσω νέων εκδόσεων και με βασικό εργαλείο τα Python Enhancement Proposals (PEPs), τα οποία είναι τυποποιημένα κείμενα που περιέχουν γενικές πληροφορίες, οδηγίες, προτάσεις και περιγραφές για τυχόν νέα χαρακτηριστικά της γλώσσας. Μάλιστα, όταν ο ίδιος έγραφε το βιβλίο του, το 2008, αναφέρει ότι:

«[...] Η σειρά 2.χ με πιο πρόσφατη έκδοση παραγωγής την 2.7.9 και η νεότερη σειρά 3.χ με πιο πρόσφατη έκδοση παραγωγής την 3.4.3. Η Python είναι κατά το μεγαλύτερο μέρος η ίδια και στις δύο σειρές εκδόσεων, αλλά με διαφορές σε πολλές λεπτομέρειες, κυρίως στη λειτουργία των ενσωματωμένων αντικειμένων όπως είναι τα λεξικά και οι συμβολοσειρές. Επίσης, στη νεότερη έκδοση έχουν αφαιρεθεί τα απαξιωμένα χαρακτηριστικά και έχει αναδιοργανωθεί η πρότυπη βιβλιοθήκη. Πολλές εφαρμογές τρίτων κατασκευαστών είναι γραμμένες σε εκδόσεις της σειράς 2.χ. Αν ξεκινάτε τώρα με την Python και δεν σας ενδιαφέρει ιδιαίτερα κάποια από αυτές τις εφαρμογές, τότε καλύτερα να ασχοληθείτε με την πιο πρόσφατη έκδοση της σειράς 3.χ.»

3.3 ΔΙΕΡΜΗΝΕΥΤΗΣ

Για να εκτελεστούν διαδραστικά (interactively) μεμονωμένες εντολές ή προγράμματα είναι απαραίτητη η εγκατάσταση του διερμηνευτή της Python, ο οποίος είναι ελεύθερα διαθέσιμος για κατέβασμα από τον επίσημο ιστότοπο της, ο οποίος είναι www.python.org. Για Microsoft Windows υπάρχουν εκδόσεις των 32 ή 64 bits. Στα λειτουργικά συστήματα Linux και Mac OS X συνηθίζεται να είναι προεγκατεστημένη, πιθανότατα όμως μία παλαιότερη έκδοσή της.

3.4 IDLE

Στην παρούσα υποενότητα, παρέχεται μια ανάλυση για το λογισμικό ανάπτυξης εφαρμογών, το οποίο χρησιμοποιεί η Python. Πιο συγκεκριμένα, για τη συγγραφή

προγραμμάτων είναι απαραίτητος ένας κειμενογράφος ή ακόμα καλύτερα ένα ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment - IDE), το οποίο είναι ένα ειδικό λογισμικό για την ανάπτυξη εφαρμογών. Η Python έρχεται μαζί με ένα εύχρηστο περιβάλλον ανάπτυξης με την ονομασία IDLE. Τα αρχικά του έρχονται από τις λέξεις Interactive DeveLopment Environment και είναι γραμμένο σε Python από τον Γκίντο βαν Ρόσσουμ. Χρησιμοποιεί τη βιβλιοθήκη γραφικών Tkinter, οπότε μπορεί να εκτελεσθεί σε περιβάλλον Linux, Windows και Mac OS X. Συγκεκριμένα ο Λεβεντέας Δ. (2008) αναφέρει ότι:

« [...] Το IDLE μας δίνει τη δυνατότητα να χρησιμοποιήσουμε διαδραστικά τον διερμηνευτή της γλώσσας, να γράψουμε και να επεξεργαστούμε προγράμματα, να τα αποθηκεύσουμε σε αρχεία, να τα εκτελέσουμε, να κάνουμε αποσφαλμάτωση. Για τη διευκόλυνση του έργου μας ως προγραμματιστές, η Python μας προσφέρει και ένα ολοκληρωμένο περιβάλλον ανάπτυξης εφαρμογών (IDE). Το περιβάλλον αυτό ονομάζεται IDLE (Integrated DeveLopment Environment) και μάλλον προήλθε τιμητικά από το όνομα ενός μέλους της ομάδας της κωμικής σειράς Monty Python's Flying Circus, τον Eric Idle.»

Συνεχίζει λέγοντας ότι το IDLE είναι ένα απλό περιβάλλον ανάπτυξης, ειδικά για νέους προγραμματιστές, στο οποίο μπορούμε να γράψουμε και να εκτελέσουμε τα προγράμματά μας. Διαθέτει τα εξής μέρη: το παράθυρο Python Shell, το παράθυρο συντάκτη προγράμματος (editor) και τον αποσφαλματωτή (debugger). Πιο συγκεκριμένα:

- **Το παράθυρο Python Shell** (κονσόλα του διερμηνευτή της Python) προορίζεται για διαδραστική και άμεση εκτέλεση εντολών, δηλαδή με την εμφάνιση της προτροπής >>> για πληκτρολόγηση εντολής και εκτέλεσής της από τον διερμηνευτή. Τα διάφορα τμήματα μιας εντολής χρωματίζονται αναλόγως. Η επισήμανση αυτή βοηθάει πάρα πολύ στη σωστή συγγραφή των εντολών. Το παράθυρο αυτό υποστηρίζει, επίσης, επανάκληση των προηγούμενων εντολών που έχουν δοθεί. Η πραγματοποίηση αυτού, συμβαίνει με ένα κλικ σε προηγούμενη εντολή και πατώντας Enter.
- **Το παράθυρο συντάκτη προγράμματος** (editor), υποστηρίζει πολλές δυνατότητες, όπως χρωματική επισήμανση των εντολών, πολλαπλές αναιρέσεις,

κατάλληλη στοίχιση των εντολών και αυτόματη συμπλήρωση εντολών. Σε ένα τέτοιο παράθυρο γράφουμε πρώτα ολόκληρο το πρόγραμμά μας (ο τρόπος αυτός ονομάζεται `program mode`), στη συνέχεια το αποθηκεύουμε σε αρχείο και στο τέλος το εκτελούμε. Τα αποτελέσματα της εκτέλεσης τα βλέπουμε στο Python Shell.

- **Ο αποσφαλματοτής (debugger)** παρακολουθεί την εκτέλεση των προγραμμάτων μας με σκοπό την εύρεση σφαλμάτων.

3.5 ΔΟΜΗ ΚΑΙ ΑΝΑΠΤΥΞΗ

Η γλώσσα χρησιμοποιεί μεταγλωττιστή (compiler) για την δημιουργία του εκτελέσιμου κώδικα και σχετίζεται με τις γλώσσες προγραμματισμού Tcl, Perl, Scheme, Java και Ruby, καθώς και με την ABC, η οποία υπήρξε η αρχική πηγή έμπνευσης για τη δημιουργία της.

3.5.1 ΟΝΟΜΑΤΟΔΟΣΙΑ ΜΕΤΑΒΛΗΤΩΝ

Τα ονόματα που λαμβάνουν οι μεταβλητές υπόκεινται στους εξής κανόνες:

- Είναι λέξεις που αποτελούνται από αλφαριθμητικούς χαρακτήρες (γράμματα και αριθμούς) και την κάτω παύλα (underscore, "_"). Μπορεί να αρχίζει με αλφαβητικό χαρακτήρα ή κάτω παύλα (_).
- Δεν επιτρέπεται να αρχίζει με αριθμό (πχ. `var4_1` είναι σωστό, αλλά το `4var_1` είναι λάθος).
- Υπάρχει διάκριση πεζών και κεφαλαίων γραμμάτων (case sensitivity), οπότε και οι μεταβλητές διαφοροποιούνται λόγω πεζών ή κεφαλαίων (πχ. άλλη η μεταβλητή `Var` και άλλη η `var`).
- Δεν επιτρέπονται δεσμευμένες λέξεις (λέξεις-κλειδιά, keywords), δηλαδή λέξεις με ιδιαίτερη σημασία (identifiers), όπως `print`, `type`, `False`, `float`, `None`, κλπ.

3.5.2 ΤΥΠΟΙ ΔΕΔΟΜΕΝΩΝ

Οι τύποι δεδομένων της Python διακρίνονται στους αρχέγονους (primitive data type), στις πιο σύνθετες δομές δεδομένων (λίστες, πλειάδες, σύνολα, λεξικά) και, λόγω των αντικειμενοστρεφών δυνατοτήτων της, στις κλάσεις (classes). Λόγω του ότι η Python είναι δυναμική γλώσσα δεν δηλώνονται οι τύποι των μεταβλητών πριν χρησιμοποιηθούν. Ο τύπος που τους αποδίδεται εξαρτάται από την τιμή που θα πάρουν κατά την διάρκεια της εκτέλεσης του προγράμματος. Όταν χρησιμοποιηθεί (πχ. σε έκφραση ή συνάρτηση) πρέπει να έχει πάρει τιμή γιατί ο διερμηνευτής (interpreter) δίνει λάθος που το ονομάζει `NameError`. Γι' αυτόν τον λόγο, η Python υποστηρίζει τον ιδιαίτερο τύπο `NoneType`.

3.5.3 ΑΡΧΕΓΟΝΟΙ ΤΥΠΟΙ ΔΕΔΟΜΕΝΩΝ

Σε αυτή την υποενότητα, αναφέρονται οι αρχέγονοι τύποι δεδομένων, οι οποίοι είναι οι ακόλουθοι:

- **int** (ακέραιος). Στην Python οι ακέραιοι μπορούν να είναι οσοδήποτε μεγάλοι.
- **float** (κινητής υποδιαστολής)
- **str** (συμβολοσειρά ή string). Είναι ακολουθία χαρακτήρων που περικλείονται από μονά ή διπλά εισαγωγικά. Σε ειδικές περιπτώσεις μπορεί να γίνει χρήση τριπλών εισαγωγικών (μονών `'''` ή διπλών `"""`).
- **bool**. Ο λογικός τύπος (Boolean) μεταβλητής με τιμές `True` ή `False`.
- **NoneType**. Αυτός ο τύπος μεταβλητής παίρνει την τιμή `None` με ανάθεση (assignment, `= None`) και αντιμετωπίζεται, στον σχεδιασμό του προγράμματος, σαν μεταβλητή που δεν έχει τιμή. `None` επιστρέφουν και οι συναρτήσεις που δεν επιστρέφουν τιμή (δεν έχουν `return <έκφραση>`), όπως για παράδειγμα η `print()`. Δεν έχει μεταβλητή τύπου `Character` (όπως για παράδειγμα η C/C++). Σε αυτή την περίπτωση χρησιμοποιείται η μεταβλητή `String` με ένα χαρακτήρα. Η συνάρτηση `type()` μας επιστρέφει τον τύπο της μεταβλητής. Αν η μεταβλητή δεν έχει πάρει τιμή μας επιστρέφει `NoneType`.

3.5.4 ΛΙΣΤΑ

Στο σημείο αυτό, θεωρήθηκε απαραίτητο να αναλυθεί ένα ακόμη χαρακτηριστικό της Python, η λίστα. Σύμφωνα με τον Φυτίλη Γ. (2017), η λίστα υλοποιεί τον γνωστό αφηρημένο τύπο δεδομένων. Είναι παρόμοια με την δομή του πίνακα (array), αλλά σε αντίθεση με αυτόν μπορεί να περιέχει ανομοιογενή αντικείμενα (items) ή αλλιώς στοιχεία (elements), τα οποία μπορεί να είναι οποιαδήποτε δομή δεδομένων (Φυτίλης Γ., 2017). Όπως στους πίνακες τα στοιχεία είναι διατεταγμένα σε συγκεκριμένη σειρά, προσπελάσιμα με δείκτες και επιτρέπεται η επανάληψη τους. Ορίζεται με ένα ζευγάρι τετραγωνισμένων αγκύλων [] και με τα στοιχεία να χωρίζονται μεταξύ τους με κόμμα, πχ. η ακολουθία [<στοιχείο1>, ..., <στοιχείοN>] ορίζει μια λίστα N στοιχείων. Μπορεί να οριστεί μια κενή λίστα μόνο με τις αγκύλες και χωρίς περιεχόμενο (Φυτίλης Γ., 2017). Επειδή η συμβολοσειρά (string) αποτελεί μια διατεταγμένη ακολουθία χαρακτήρων μοιάζει με λίστα και χρησιμοποιεί παρόμοιους τελεστές (operators) και συναρτήσεις (methods). Η μεγάλη διαφορά τους είναι ότι η λίστα είναι δυναμικός ή αλλιώς μεταβλητός (mutable) τύπος δεδομένων, ενώ η συμβολοσειρά στατικός ή αλλιώς αμετάβλητος (immutable) τύπος δεδομένων. Στη λίστα μπορούμε να αλλάξουμε τα στοιχεία που περιέχει. Στη συμβολοσειρά η αλλαγή των χαρακτήρων που περιέχει γίνεται μόνο με αντικατάσταση (overwriting) μέσω ανάθεσης (assignment) από νέα συμβολοσειρά (Φυτίλης Γ., 2017).

3.5.5 ΠΛΕΙΑΔΑ

Ένα επιπλέον στοιχείο της Python, είναι η πλειάδα²⁰. Η πλειάδα υλοποιεί την αντίστοιχη μαθηματική έννοια της πλειάδας. Είναι μια πεπερασμένη ακολουθία στοιχείων διατεταγμένων σε συγκεκριμένη σειρά και μοιάζει πολύ με την λίστα (list). Ορίζεται με ένα ζευγάρι παρενθέσεων () και με τα στοιχεία να χωρίζονται μεταξύ τους με κόμμα, πχ. η ακολουθία (<στοιχείο1>, ..., <στοιχείοN>) ορίζει μια πλειάδα N στοιχείων. Αν το επιτρέπουν τα συμφραζόμενα μπορούν να παραληφθούν οι παρενθέσεις. Διαφέρει από την δομή της λίστας στο ότι η πλειάδα είναι στατική ή αλλιώς

²⁰ [https://el.wikipedia.org/wiki/Python#Πλειάδα_\(tuple\)](https://el.wikipedia.org/wiki/Python#Πλειάδα_(tuple))

αμετάβλητη (immutable) δομή δεδομένων. Αφού δημιουργηθεί δεν μπορεί να μεταβληθεί. Η χρήση της πλειάδας είναι προτιμότερη από της λίστας όταν δεν χρειάζεται να αλλάξουμε τα στοιχεία της γιατί θεωρητικά είναι αποτελεσματικότερη σε ταχύτητα στις διάφορες λειτουργίες της. Η πλειάδα που περιέχει ένα στοιχείο ορίζεται πάντα με ένα κόμμα μετά το στοιχείο, πχ. (<στοιχείο>,), και αυτό γιατί οι παρενθέσεις χρησιμοποιούνται και στην ομαδοποίηση των εκφράσεων (expressions) και η γραφή χωρίς κόμμα, πχ. = (<στοιχείο>) λαμβάνεται από τον διερμηνευτή (interpreter) ως ανάθεση (assignment), δηλαδή = <στοιχείο>.

3.5.6 ΛΕΞΙΚΟ

Το λεξικό (dictionary) είναι μια δομή δεδομένων για την αποθήκευση ζευγαριών τιμών της μορφής κλειδί : τιμή (key : value). Ορίζεται με ένα ζευγάρι αγκύλων {} και με τα ζευγάρια να χωρίζονται μεταξύ τους με κόμμα, πχ. η ακολουθεία

ορίζει ένα λεξικό N ζευγαριών. Ανάμεσα στο κλειδί και την τιμή υπάρχει η άνω-κάτω τελεία (:). Τα κλειδιά πρέπει να είναι μοναδικά γιατί χρησιμεύουν στην προσπέλαση της αντίστοιχης τιμής. Τα κλειδιά πρέπει να είναι στατικοί ή αλλιώς αμετάβλητοι (immutable) τύποι δεδομένων, όπως ακέραιοι αριθμοί, συμβολοσειρές, αμετάβλητες πλειάδες (με την προϋπόθεση όλα τα στοιχεία της πλειάδας να είναι αμετάβλητα).

3.5.7 ΑΝΑΘΕΣΗ ΤΙΜΩΝ

Επειδή η Python είναι δυναμική γλώσσα προγραμματισμού (dynamically typed) διαφέρει από τις άλλες παραδοσιακές γλώσσες προστακτικού προγραμματισμού (Imperative programming) στον τρόπο που γίνεται η ανάθεση τιμών στις μεταβλητές της. Συνήθως οι μεταβλητές δηλώνονται πριν τους ανατεθούν τιμές και μαζί δηλώνεται και ο τύπος τους (integer, floating point, character, κλπ.). Στην Python οι μεταβλητές δηλώνονται μαζί με την ανάθεση των τιμών σε αυτές (ακόμα και όταν τους δοθεί η τιμή None) και δεν είναι δεσμευμένες σε κάποιο συγκεκριμένο τύπο δεδομένων. Μία

μεταβλητή μπορεί να είναι ακεραία (integer) και η ίδια στη συνέχεια να γίνει συμβολοσειρά (string) αν της ανατεθεί τιμή συμβολοσειράς. Στην Python, όταν κατά την εκτέλεση ενός προγράμματος, αναθέτουμε τιμή σε μια μεταβλητή x, πχ. `x = 2`, η τιμή 2 αντιγράφεται σε χώρο μνήμης που δεσμεύεται (memory allocation), μαζί με άλλες ιδιότητες, όπως πχ. ότι ο τύπος της είναι ακέραιος (integer). Στην μεταβλητή x αποδίδεται η διεύθυνση (memory address) αυτής της θέσης στην μνήμη. Η μεταβλητή λειτουργεί σαν δείκτης (pointer). Η ίδια η μεταβλητή δεν έχει κάποια τιμή ή τύπο δεδομένων, αλλά έχει τη θέση της μνήμης που «αντιπροσωπεύει». Σε ένα πρόγραμμα μετά την ανάθεση `x = 2` (η μεταβλητή είναι τύπου integer), στην συνέχεια της εκτέλεσης μπορεί να εκτελεστεί η ανάθεση `x = "hellow world"`, οπότε η μεταβλητή «αντιπροσωπεύει» την συμβολοσειρά "hellow world" (είναι τύπου string). Στην δεύτερη ανάθεση η τιμή 2 δεν «αντιπροσωπεύεται» από κάποια μεταβλητή (δεν χρειάζεται πλέον στο πρόγραμμα) και ο συλλέκτης απορριμμάτων (garbage collector), της Python, αναλαμβάνει «αυτόματα» να αποδεσμεύσει (memory de-allocation) τον χώρο της μνήμης που αυτή κατέχει.

3.5.8 ΕΣΟΧΕΣ

Η στοίχιση στα αριστερά ή αλλιώς οι αριστερές εσοχές (indentation) έχουν μεγάλη σημασία γιατί ορίζουν τα block (ομαδοποίηση) του κώδικα. Αυτό σε άλλες γλώσσες, όπως C και Java επιτυγχάνεται με την χρήση των αγκύλων { }. Στην Python το block κώδικα αρχίζει εκεί που η αριστερή εσοχή αυξάνεται προς τα δεξιά και τελειώνει εκεί που καταργείται η εσοχή. Εσοχή μέσα σε εσοχή σημαίνει block μέσα σε block (nested blocks). Η εσοχή μπορεί να είναι οσοδήποτε μεγάλη ή μικρή, αλλά συνίσταται η χρήση 4 κενών χαρακτήρων για ευκολότερη αναγνωσιμότητα του κώδικα. Για παράδειγμα μια εμφωλιασμένη εντολή if ορίζεται ως εξής:

```
if <λογική έκφραση1> :  
    εντολή1  
    εντολή2  
    if <λογική έκφραση2> :  
        εντολή3
```

```
        εντολή4
    εντολή5
εντολή6
```

Το block της πρώτης if περιέχει τις εντολή1, εντολή2, την δεύτερη if και τελειώνει με την εντολή5. Η εντολή6 είναι εκτός block (δεν ανήκει στην εξωτερική if). Η εσωτερική if (εμφωλιασμένη) με την δική της εσοχή ορίζει ένα block μέσα στο block της εξωτερικής if.

3.5.9 ΤΕΛΕΣΤΕΣ

Οι τελεστές (operators) είναι λειτουργίες που κάνουν κάτι και μπορούν να αναπαρασταθούν με σύμβολα όπως το + ή με λέξεις κλειδιά όπως το and. Η αποτίμηση μιας έκφρασης παράγει μία τιμή και αυτός είναι και ο λόγος που μία έκφραση μπορεί να βρίσκεται στο δεξί μέρος μια εντολής εκχώρησης. Όταν μία μεταβλητή εμφανίζεται σε έκφραση, αντικαθίσταται από την τιμή της, προτού αποτιμηθεί η έκφραση. Εάν εισάγετε μία έκφραση στον διερμηνευτή, αυτός την υπολογίζει και δείχνει το αποτέλεσμα.

Σειρά προτεραιότητας τελεστών

Ας πούμε ότι έχουμε μία έκφραση όπως $2 + 3 * 4$. Ποιο γίνεται πρώτα, η πρόσθεση ή ο πολλαπλασιασμός; Τα μαθηματικά μας λένε ότι πρώτα πρέπει να γίνει ο πολλαπλασιασμός. Αυτό σημαίνει ότι ο τελεστής του πολλαπλασιασμού έχει υψηλότερη προτεραιότητα από τον τελεστή της πρόσθεσης. Ο ακόλουθος πίνακας μας δίνει τον πίνακα προτεραιοτήτων για την Python, από τη χαμηλότερη προς την υψηλότερη προτεραιότητα. Αυτό σημαίνει ότι σε μία δεδομένη έκφραση, η Python θα υπολογίσει τους τελεστές και τις εκφράσεις που βρίσκονται χαμηλότερα στον πίνακα πριν από αυτούς που βρίσκονται ψηλότερα. Είναι πολύ καλύτερο να χρησιμοποιείτε παρενθέσεις για να ομαδοποιείτε σωστά τελεστές και τελεστέους ώστε να ορίσετε ξεκάθαρα την προτεραιότητα. Έτσι το πρόγραμμα γίνεται πιο ευανάγνωστο. Ο πίνακας που ακολουθεί,

παρουσιάζει την προτεραιότητα των τελεστών στην γλώσσα προγραμματισμού της Python.

Τελεστής	Περιγραφή
<code>lambda</code>	Έκφραση Lambda
<code>or</code>	Λογικό Ή
<code>and</code>	Λογικό ΚΑΙ
<code>not x</code>	Λογικό ΟΧΙ
<code>in, not in</code>	Έλεγχοι συμμετοχής
<code>is, is not</code>	Έλεγχοι ταυτότητας
<code><, <=, >, >=, !=, ==</code>	Συγκρίσεις
<code> </code>	Δυαδικό Ή
<code>^</code>	Δυαδικό αποκλειστικό Ή
<code>&</code>	Δυαδικό ΚΑΙ
<code><<, >></code>	Μεταθέσεις
<code>+, -</code>	Πρόσθεση και αφαίρεση
<code>*, /, //, %</code>	Πολλαπλασιασμός, Διαίρεση, Διαίρεση στρογγυλοποιημένη προς τα κάτω, Υπόλοιπο
<code>+x, -x</code>	Θετικό, Αρνητικό
<code>~x</code>	Δυαδικό ΟΧΙ
<code>**</code>	Ύψωση σε δύναμη
<code>x.attribute</code>	Αναφορά σε χαρακτηριστικό
<code>x[index]</code>	Subscription
<code>x[index1:index2]</code>	Κομμάτισμα (Slicing)
<code>f(arguments ...)</code>	Κλήση συνάρτησης
<code>(expressions, ...)</code>	Συνένωση ή εμφάνιση πλειάδας
<code>[expressions, ...]</code>	Προβολή λίστας
<code>{key:datum, ...}</code>	Προβολή λεξικού

Εικόνα 6. Τελεστές στην Python

Τελεστής	Όνομα	Εξήγηση	Παραδείγματα
+	Συν	Προσθέτει δύο αντικείμενα.	To <code>3 + 5</code> δίνει <code>8</code> . To <code>'a' + 'b'</code> δίνει <code>'ab'</code> .
-	Μείον	Είτε δίνει έναν αρνητικό αριθμό, ή αφαιρεί έναν αριθμό από έναν άλλο.	To <code>-5.2</code> δίνει έναν αρνητικό αριθμό. To <code>50 - 24</code> δίνει <code>26</code> .
*	Επί	Δίνει το γινόμενο δύο αριθμών ή μία συμβολοσειρά (string) επαναλαμβανόμενη τόσες φορές.	To <code>2 * 3</code> δίνει <code>6</code> . To <code>'la' * 3</code> δίνει <code>'lalala'</code> .
**	Δύναμη	Επιστρέφει το x υψωμένο στη δύναμη y.	To <code>3 ** 4</code> δίνει <code>81</code> (δηλαδή <code>3 * 3 * 3 * 3</code>).
/	Διά	Διαιρεί το x με το y.	To <code>4 / 3</code> δίνει <code>1.3333333333333333</code> .
//	Διαίρεση στρογγυλοποιημένη προς τα κάτω (Floor Division)	Επιστρέφει τον κοντινότερο (προς τα κάτω) ακέραιο στο πηλίκο.	To <code>4 // 3</code> δίνει <code>1</code> .
%	Υπόλοιπο	Επιστρέφει το υπόλοιπο της διαίρεσης.	To <code>8 % 3</code> δίνει <code>2</code> . To <code>-25.5 % 2.25</code> δίνει <code>1.5</code> .
<<	Αριστερή μετάθεση	Μεταθέτει τα δυαδικά ψηφία (bits) του αριθμού προς τα αριστερά κατά το πλήθος των θέσεων που καθορίστηκε. (Κάθε αριθμός αναπαρίστανται στη μνήμη με δυαδικά ψηφία (bits, binary digits) -δηλαδή με 0 και 1).	To <code>2 << 2</code> δίνει <code>8</code> . Το 2 αναπαρίστανται ως σε bits ως <code>10</code> . Η μετάθεση προς τα αριστερά κατά 2 bits μας δίνει <code>1000</code> που παριστάνει το δεκαδικό <code>8</code> .
>>	Δεξιά μετάθεση	Μεταθέτει τα bits του αριθμού προς τα δεξιά κατά το πλήθος των θέσεων που καθορίστηκε.	To <code>11 >> 1</code> δίνει <code>5</code> . Το 11 αναπαρίστανται σε bits ως <code>1011</code> που όταν μετατεθούν δεξιά κατά 1 bit μας δίνει <code>101</code> το οποίο είναι το δεκαδικό <code>5</code> .
&	Δυαδικό ΚΑΙ	Δυαδικό ΚΑΙ των αριθμών.	To <code>5 & 3</code> δίνει <code>1</code> .
	Δυαδικό Ή	Δυαδικό Ή των αριθμών.	To <code>5 3</code> δίνει <code>7</code> .
^	Δυαδικό αποκλειστικό Ή	Δυαδικό αποκλειστικό Ή των αριθμών.	To <code>5 ^ 3</code> δίνει <code>6</code> .

Εικόνα 7. Τελεστές στην Python

~	Δυαδική αντιστροφή	Το δυαδικό αντίστροφο του x είναι -(x+1).	To <code>~5</code> δίνει <code>-6</code> .
<	Μικρότερο από	Επιστρέφει το αν το x είναι μικρότερο από το y. Όλοι οι τελεστές σύγκρισης επιστρέφουν True (Αληθής) ή False (Ψευδής). Σημειώστε ότι τα ονόματα αυτά ξεκινούν με κεφαλαίο.	To <code>5 < 3</code> δίνει <code>False</code> και το <code>3 < 5</code> δίνει <code>True</code> . Οι συγκρίσεις μπορούν να συνδυαστούν αλυσιδωτά κατά βούληση: Το <code>3 < 5 < 7</code> δίνει <code>True</code> .
>	Μεγαλύτερο από	Επιστρέφει το αν το x είναι μεγαλύτερο από το y.	To <code>5 > 3</code> επιστρέφει <code>True</code> . Αν και οι δύο τελεστές είναι αριθμοί, πρώτα μετατρέπονται σε έναν κοινό τύπο. Αλλιώς, επιστρέφει πάντα <code>False</code> .
<=	Μικρότερο ή ίσο	Επιστρέφει το αν το x είναι μικρότερο από ή ίσο με το y.	To <code>x = 3; y = 6; x <= y</code> επιστρέφει <code>True</code> .
>=	Μεγαλύτερο ή ίσο	Επιστρέφει το αν το x είναι μεγαλύτερο από ή ίσο με το y.	To <code>x = 4; y = 3; x >= 3</code> επιστρέφει <code>True</code> .
=	Ίσο	Συγκρίνει αν τα αντικείμενα είναι ίσα.	To <code>x = 2; y = 2; x == y</code> επιστρέφει <code>True</code> . To <code>x = 'str'; y = 'str'; x == y</code> επιστρέφει <code>False</code> . To <code>x = 'str'; y = 'str'; x == y</code> επιστρέφει <code>True</code> .
!=	Διαφορετικό	Συγκρίνει αν τα αντικείμενα ΔΕΝ είναι ίσα.	To <code>x = 2; y = 3; x != y</code> επιστρέφει <code>True</code> .
not	Λογικό ΟΧΙ	Αν το x είναι True, επιστρέφει False. Αν το x είναι False, επιστρέφει True.	To <code>x = True; not x</code> επιστρέφει <code>False</code> .
and	Λογικό ΚΑΙ	Το x and y επιστρέφει False αν το x είναι False, αλλιώς επιστρέφει υπολογίζει και επιστρέφει την τιμή του y.	To <code>x = False; y = True; x and y</code> επιστρέφει <code>False</code> αφού το x είναι <code>False</code> . Σε αυτή την περίπτωση, η Python δε θα ελέγξει την τιμή του y αφού γνωρίζει ότι η αριστερή πλευρά της έκφρασης and είναι <code>False</code> που υποδηλώνει ότι ολόκληρη η έκφραση θα είναι <code>False</code> ανεξάρτητα από τις άλλες τιμές. Αυτή η τεχνική αποκαλείται <i>short-circuit evaluation</i> .
or	Λογικό Ή	Αν το x είναι True, επιστρέφει True, αλλιώς υπολογίζει και επιστρέφει την τιμή του y.	To <code>x = True; y = False; x or y</code> επιστρέφει <code>True</code> . Η <i>short-circuit evaluation</i> εφαρμόζεται και εδώ.

Εικόνα 8. Τελεστές στην Python

1. Τελεστές πράξεων

Υποστηρίζουν τους γνωστούς, με τις άλλες γλώσσες προγραμματισμού, τελεστές (operators), όπως τους +, -, *, /, %, που όμως έχουν κάποιες διαφοροποιήσεις ανάλογα με τους τελεστέους (operands), δηλαδή τον τύπο των μεταβλητών, όπου εφαρμόζονται. Ο τελεστής της πρόσθεσης + όταν οι τελεστέοι είναι Strings εκτελεί την πράξη της συνένωσης (concatenation) και ο πολλαπλασιασμός * την πράξη της αναπαραγωγής (replication) του String όταν αυτό πολλαπλασιάζεται με ακέραιο (integer). Δηλαδή η έκφραση (expression): "abc" + "def" θα δώσει "abcdef" και η "abc" * 3 ή 3 * "abc" θα δώσει "abcabcabc" αντίστοιχα.

2. Τελεστές λογικής

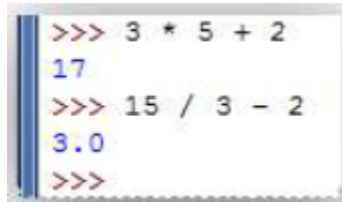
Υποστηρίζει τους λογικούς τελεστές and, or και not για μεταβλητές τύπου Boolean. Τα εμφωλιασμένα (nested) not αποδίδονται ως διπλή άρνηση (άρνηση στην άρνηση). Η σειρά (προτεραιότητα) εκτέλεσης είναι not, and, or.

Παρακάτω, δίνονται μερικά απλά παραδείγματα προτεραιότητας τελεστών:

```
>>> 2 * (4 - 1)
6
>>> 2 * 4 - 1
7
>>> (2 + 1) ** (5 - 2)
27
>>> 2 ** 1 + 1
3
>>> 2 ** (1 + 1)
4
>>>
```

Εικόνα 9. Προτεραιότητα τελεστών

Ο πολλαπλασιασμός και η διαίρεση έχουν την ίδια προτεραιότητα, η οποία είναι μεγαλύτερη από την πρόσθεση και την αφαίρεση που μεταξύ τους έχουν επίσης την ίδια προτεραιότητα. Για παράδειγμα:



```
>>> 3 * 5 + 2
17
>>> 15 / 3 - 2
3.0
>>>
```

Εικόνα 10. Προτεραιότητα τελεστών

Τελεστές με την ίδια προτεραιότητα αποτιμώνται από τα αριστερά προς τα δεξιά.

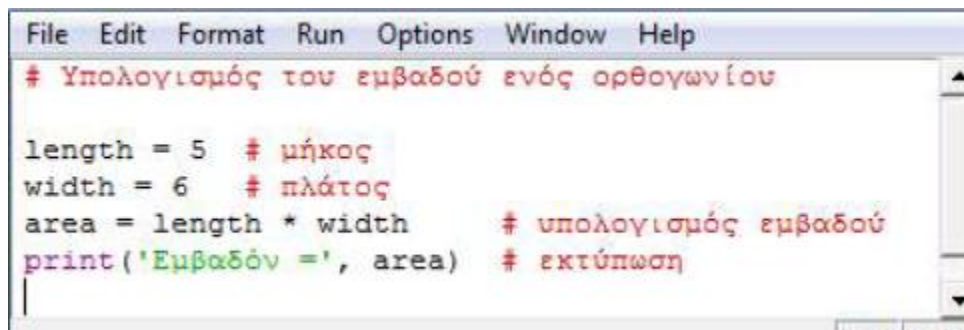
3.5.10 ΕΜΒΕΛΕΙΑ ΟΝΟΜΑΤΩΝ ΟΝΤΟΤΗΤΩΝ

Στις οντότητες (μεταβλητές, συναρτήσεις, μέθοδοι, κλπ), που δημιουργούνται σε ένα πρόγραμμα, για να μπορούν να χρησιμοποιηθούν (προσπελαστούν), αποδίδονται ονόματα (name binding). Για παράδειγμα στην Python μια οντότητα, όπως είναι μια μεταβλητή, δεν δηλώνεται, αλλά δημιουργείται και ονοματίζεται όταν της γίνει ανάθεση (assignment) κάποιας τιμής. Οντότητες που δημιουργούνται μέσα σε ένα block κώδικα, όπως το σώμα μίας συνάρτησης, μίας εντολής if ή for, κλπ, έχουν περιορισμένη εμβέλεια και μπορούν να χρησιμοποιηθούν μόνο μέσα στο block που έχουν ονομαστεί. Ένα τέτοιο όνομα δεν είναι προσπελάσιμο εκτός του block ακόμα και αν έχει την ίδια ονομασία και λέγεται ότι είναι τοπικής εμβέλειας (local scope). Ένα όνομα που δηλώνεται στο κύριο σώμα (block) του προγράμματος μπορεί να προσπελαστούν σε ολόκληρο το πρόγραμμα, ακόμα και στα block που έχουν οριστεί μέσα στο πρόγραμμα και λέγεται ότι είναι καθολικής εμβέλειας (global scope). Μέσα σε ένα block ο διερμηνευτής (interpreter) χρησιμοποιεί την τοπική ονομασία μιας οντότητας ακόμα και αν υπάρχει άλλη με την ίδια ονομασία και καθολική εμβέλεια. Συγκεκριμένα ο διερμηνευτής όταν συναντήσει ένα όνομα (πχ. μεταβλητής) μέσα σε ένα block ελέγχει πρώτα αν έχει δημιουργηθεί αυτό το όνομα μέσα στο block (οπότε είναι τοπικής εμβέλειας) και το χρησιμοποιεί, έστω και αν υπάρχει ίδια ονομασία εκτός του block (καθολικής εμβέλειας). Αν δεν το βρει μέσα

στο block τότε και μόνο τότε ελέγχει αν υπάρχει τέτοια ονομασία που να έχει δημιουργηθεί στο κύριο μέρος του προγράμματος (είναι καθολικής εμβέλειας) και χρησιμοποιεί αυτήν. Δηλαδή μια καθολικής εμβέλειας μεταβλητή χρησιμοποιείται μέσα σε ένα block, μόνο αν δεν υπάρχει τοπικής εμβέλειας μεταβλητή με το ίδιο όνομα, μέσα στο block.

3.5.11 ΣΧΟΛΙΑ

Καθώς γράφουμε όλο και μεγαλύτερα προγράμματα, μεγαλώνει και η δυσκολία να καταλαβαίνουμε αυτά που γράφουμε. Πολύ περισσότερο όμως, αν προσπαθήσει να τα διαβάσει κάποιος τρίτος. Γενικά, οι γλώσσες προγραμματισμού συμπυκνώνουν νοήματα και γι' αυτό είναι πολλές φορές δύσκολο να τις διαβάσει κάποιος και να καταλάβει τι θέλει να κάνει ένα κομμάτι κώδικα. Χρειάζεται λοιπόν να προσθέτουμε σχόλια (comments) στα προγράμματα που γράφουμε. Τα σχόλια στην Python αρχίζουν πάντα με τον χαρακτήρα #. Οτιδήποτε ακολουθεί μετά το # αγνοείται από την Python μέχρι το τέλος της γραμμής (προορίζεται για ανάγνωση μόνο από άνθρωπο). Ας δούμε ένα παράδειγμα:



Εικόνα 11. Παράδειγμα σχολίων

3.5.12 ΔΗΛΩΣΗ GLOBAL

Όταν μέσα σε ένα block γίνει η ανάθεση `num = 123`, τότε δημιουργείται (αναγκαστικά) η τοπικής εμβέλειας μεταβλητή `num` ακόμα και αν υπάρχει καθολικής εμβέλειας `num`. Στην περίπτωση που δεν είναι επιθυμητό να δημιουργηθεί η τοπική `num`,

αλλά η τιμή 123 να ανατεθεί στην καθολική `num` γίνεται χρήση της δήλωσης `global`. Δηλώνοντας `global num`, πριν από την `num = 123`, ο διερμηνευτής δεν δημιουργεί τοπική `num`, αλλά δίνει την τιμή 123, στην καθολική που ήδη υπάρχει.

3.6 ΠΑΡΑΔΕΙΓΜΑ ΠΡΟΓΡΑΜΜΑΤΟΣ

Ένα από τα πιο απλά προγράμματα στην γλώσσα Python είναι η εμφάνιση ενός γραπτού αποτελέσματος (π.χ. Γεια σου, κόσμε!): `>>> print("Γεια σου, κόσμε!")` Γεια σου, κόσμε! Ένα ιδιαίτερο χαρακτηριστικό της γλώσσας είναι η χρήση κενών διαστημάτων σαν εσοχές (`whitespace`) για τον διαχωρισμό των συντακτικών δομών (`block of code`) του προγράμματος, σε αντίθεση με την πρακτική σε άλλες γλώσσες όπου για τον ίδιο σκοπό χρησιμοποιούνται ειδικά σύμβολα (πχ αγκύλες). Αυτό, σε συνδυασμό με το ότι χρησιμοποιεί πλήρεις αγγλικές λέξεις στη θέση συμβόλων, καθιστούν τον κώδικα της Python ευανάγνωστο από όσους έχουν βασική γνώση των αγγλικών.

```
age = 21
if age >= 18:
    print("You vote")
else:
    print("You don't vote")
```

Εικόνα 12. Παράδειγμα προγράμματος.

3.6.1 ΔΙΑΔΙΚΑΣΤΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Στην παρούσα υποενότητα δίνεται το εξής παράδειγμα, έστω ο παρακάτω γεννήτορας (`generator`) ο οποίος παράγει πρώτους αριθμούς:

```

from itertools import count
def generate_primes(stop_at=None):
    primes = []
    for n in count(2):
        if stop_at is not None and n > stop_at:
            return
        composite = False
        for p in primes:
            if not n % p:
                composite = True
                break
            elif p*2 > n:
                break
        if not composite:
            primes.append(n)
            yield n

```

Εικόνα 13. Παράδειγμα προγράμματος.

Και για να καλέσουμε τη συνάρτηση, υπολογίζοντας τους πρώτους αριθμούς από το 1 στο 100:

```

for i in generate_primes():
    if i > 100: break
    print(i)

```

Εικόνα 14. Παράδειγμα συνάρτησης υπολογισμού πρώτων αριθμών από 1 έως 100.

3.6.2 ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΗΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Σε αυτή την υποενότητα δίνεται ένα παράδειγμα αντικειμενοστραφούς προγράμματος γραμμένο σε Python, το οποίο παρουσιάζει μια κλάση Υπάλληλος:

```

class Employee:
    def __init__(self, id, name, position):
        self.id = id;
        self.name = name;
        self.position = position;
    def printEmployee(self):
        print(self.id, "|", self.name, "|", self.position);

e1 = Employee(1230, "Νικόλαος Παπαγεωργίου", "Βοηθός Προγραμματιστή");
e1.printEmployee();

```

Εικόνα 15. Παράδειγμα κλάσης "υπάλληλος".

ΚΕΦΑΛΑΙΟ 4: ΠΡΟΓΡΑΜΜΑ ΠΡΟΣΟΜΟΙΩΣΗΣ ΚΑΙ ΜΟΝΤΕΛΟΠΟΙΗΣΗΣ ΜΙΑΣ ΕΠΙΔΗΜΙΑΣ

4.1 ΤΟ ΠΡΟΓΡΑΜΜΑ

Έως τώρα έχει αναλυθεί επακριβώς η προσομοίωση, μοντελοποίηση, το μοντέλο SIR μαζί με όλες τις πτυχές του. Σε αυτό το κεφάλαιο θα αναλυθεί πως ακριβώς όλες αυτές οι έννοιες συνδέονται, και φέρουν ως αποτέλεσμα το πρόγραμμα για το οποίο συγγράφεται αυτή η αναφορά, το οποίο αφορά την προσομοίωση και μοντελοποίηση μιας επιδημίας. Αρχικά, το πρόγραμμα γράφθηκε σε γλώσσα προγραμματισμού Python, με την χρήση του εργαλείου PyCharm. Σκοπός του προγράμματος είναι να προσομοιώνει την εξάπλωση μιας επιδημίας πάνω σε ένα δοθέν πειραματικό πληθυσμό που όρισε ο χρήστης. Το πρόγραμμα επίσης, έχει ενσωματωμένα κάποια παθογόνα τα οποία ο χρήστης μπορεί να χρησιμοποιήσει για να προσομοιώσει την εξάπλωση του παθογόνου στο πληθυσμό σε μήκος διακύμανσης μέρες τις οποίες ο ίδιος έχει ορίσει. Στο πρόγραμμα έχουν προκαθοριστεί στις προεπιλογές τα εξής παθογόνα: τον ιλαρά το οποίο είναι υπαίτιο για την νόσο του ιλαρά, το πρωτόζωο το οποίο είναι υπαίτιο για την νόσο της ελονοσίας, τον Βάκιλο του Γερσίν, ο οποίος είναι υπαίτιος για την νόσο της πανούκλας, τον πολιοϊό, ο οποίος είναι υπαίτιος για την πολιομυελίτιδα, και τέλος ο ιός του Έμπολα, ο οποίος είναι υπαίτιος για τη γνωστή νόσο Έμπολα. Ο χρήστης με το που ξεκινά το πρόγραμμα, αλληλοεπιδρά με αυτό μέσα από μενού τα οποία τον βοηθούν να περιηγηθεί.

Αρχικά, ο χρήστης με το που ξεκινά το πρόγραμμα βλέπει την παρακάτω οθόνη.

```
Connected to pydev debugger (build 192.5728.105)
Καλώς ήλθατε!

1. Προσαρμοσμένος πληθυσμός με προεπιλεγμένο παθογόνο παράγοντα.
2. Προσαρμοσμένος πληθυσμός με προσαρμοσμένο παθογόνο παράγοντα.
3. Έξοδος.

Παρακαλώ επιλέξτε:
```

Εικόνα 16. Αρχικό μενού πλοήγησης του αλγορίθμου.

Όπως προαναφέρθηκε, ο χρήστης αλληλοεπιδρά με ένα μενού πλοήγησης στο οποίο μπορεί να επιλέξει:

1. Προσαρμοσμένο πληθυσμό με προεπιλεγμένο παθογόνο παράγοντα
2. Προσαρμοσμένο πληθυσμό με προσαρμοσμένο παθογόνο παράγοντα
3. Έξοδο και να τερματίσει το πρόγραμμα.

Στη συνέχεια, αν ο χρήστης επιλέξει την **επιλογή 1** ανακατευθύνεται στο παρακάτω.

```
Παρακαλώ επιλέξτε: 1
-----
Προεπιλεγμένα παθογόνα

Παρακαλώ επιλέξτε ένα παθογόνο της παρακάτω λίστας:

1. Ιλαράς
2. Πρωτόζωο
3. Βάκιλος του Γερσίν
4. Πολιοϊός
5. Έμπολα
Παρακαλώ επιλέξτε:
```

Εικόνα 17. Προσαρμοσμένος πληθυσμός με προεπιλεγμένο παθογόνο παράγοντα.

Εφόσον ο χρήστης έχει επιλέξει την επιλογή ένα του προηγούμενου μενού, του δίνεται η λίστα των προεπιλεγμένων παθογόνων τα οποία θέλει να προσομοιώσει σε έναν δειγματικό και πειραματικό πληθυσμό με πειραματικές συνθήκες. Έστω, για

παραδειγματικούς σκοπούς, ότι επιλέγουμε το παθογόνο του Ιλαρά (επιλογή 1). Κατά την επιλογή ο χρήστης βλέπει κάποιες περαιτέρω πληροφορίες για το παθογόνο που επέλεξε.

```
Παθογόνο που επιλέχτηκε:  
Όνομα: Ιός του Ιλαρά  
Τύπος: Ιός  
Αναγνωριστικό NCBI: 11234  
Αιτιώδης παράγοντας της "Ιλαράς"  
Μέση μολυσματική περίοδος: 8.00 ημέρες  
-----
```

Εικόνα 18. Οθόνη προβολής πληροφοριών παθογόνου.

Στη συνέχεια, το πρόγραμμα ζητά από το χρήστη να εισάγει τον αριθμό των ατόμων στο πληθυσμό, τον αριθμό των ευπαθών ατόμων, τον αριθμό των αρχικά μολυσμένων ατόμων, και τέλος τον αριθμό των ανοσοποιημένων ατόμων. Μετά την εισαγωγή, γίνεται μια συνολική αναφορά των τιμών που εισάγει ο χρήστης, όπως φαίνεται και παρακάτω.

```
-----  
Δημιουργία δοκιμαστικού πληθυσμού.  
  
Εισάγετε τον συνολικό αριθμό ατόμων στον πληθυσμό: 10000  
Εισάγετε τον αριθμό των ευπαθών ατόμων: 1000  
Εισάγετε τον αριθμό των αρχικά μολυσμένων ατόμων: 100  
Εισάγετε τον αριθμό των ανοσοποιημένων ατόμων: 2  
  
Ο δοκιμαστικός πληθυσμός αποτελείται από 1000 ευπαθείς, 100 μολυσμένους,  
και 2 ανοσοποιημένους.
```

Εικόνα 19. Οθόνη προβολής συνολικής αναφοράς εισαχθέντων τιμών.

Μετά από αυτό το βήμα, όπως φαίνεται και παρακάτω,

```
Εισάγετε το μήκος προσομοίωσης σε μέρες: 70
-----
Ημέρα 0: 100 ευπαθείς, 100 μολυσμένοι, 2 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 1: 97.8 ευπαθείς, 89.8 μολυσμένοι, 9812.4 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 2: 95.8 ευπαθείς, 80.6 μολυσμένοι, 9823.6 ανοσοποιημένοι, 10000 συνολικά.
```

Εικόνα 20. Εισαγωγή μήκους προσομοίωσης σε ημέρες.

ζητείται από το χρήστη να εισάγει το μήκος της προσομοίωσης σε ημέρες. Από εκεί και πέρα, το πρόγραμμα ξεκινάει τη προσομοίωση και για κάθε ημέρα που περνάει καταγράφει τους ευπαθείς, τους μολυσμένους, τους ανοσοποιημένους, και πόσοι άνθρωποι υπάρχουν συνολικά. Μετά τη προσομοίωση, το πρόγραμμα εμφανίζει στην οθόνη τα παρακάτω στατιστικά στοιχεία.

```
Στατιστικά στοιχεία:

Παθογόνο R0 σε πλήρως ευαίσθητο πληθυσμό: 5.0
Παθογόνος περίοδος μολυσματικότητας: 15.00 ημέρες
Όριο ευαισθησίας: 2000.0 άτομα
Ανοσολογικά άτομα που χρειάστηκαν για την πρόληψη της επιδημίας: 8000.0 άτομα
Όριο κατωφλίου: 0.2
-----

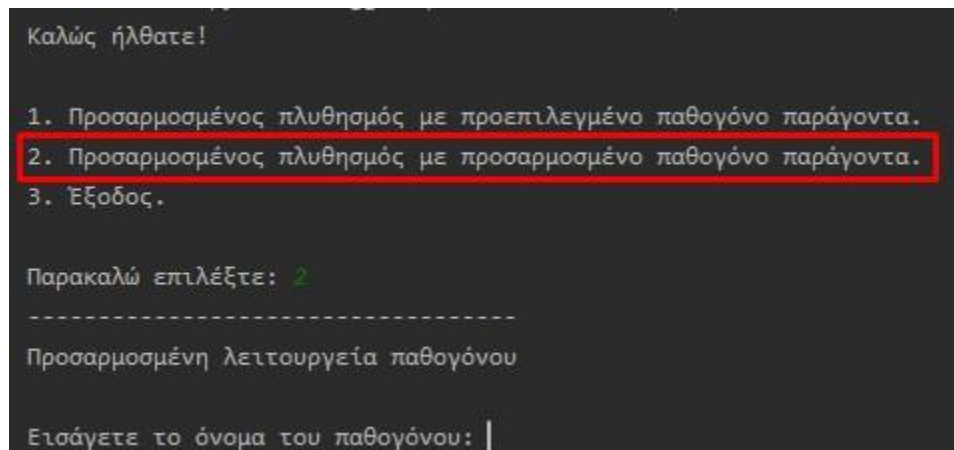
Process finished with exit code 0
```

Εικόνα 21. Αποτελέσματα αλγορίθμου και τερματισμός της προσομοίωσης.

Σε αυτό το σημείο, ο αλγόριθμος τερματίζει και η προσομοίωση ολοκληρώνεται.

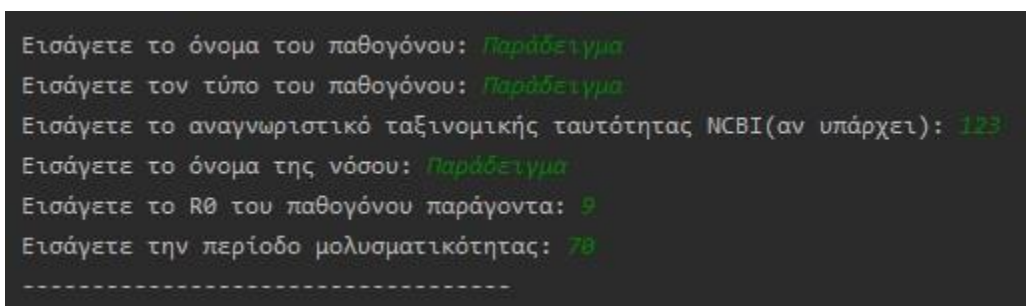
Σε περίπτωση, όμως, όπου ο χρήστης θελήσει, το πρόγραμμα δίνει την δυνατότητα να δημιουργηθεί προσαρμοσμένο παθογόνο και να προσομοιώσει την μολυσματική του περίοδο πάνω σε προσαρμοσμένο πληθυσμό με προσαρμοσμένο αριθμό ημερών. Έτσι,

όπως φαίνεται και παρακάτω, από το αρχικό μενού πλοήγησης ο χρήστης διαλέγοντας την **επιλογή 2**:



Εικόνα 22. Προσαρμοσμένος πληθυσμός με προσαρμοσμένο παθογόνο παράγοντα.

από το αρχικό μενού, το πρόγραμμα ζητάει από τον χρήστη να εισάγει το όνομα του παθογόνου, καθώς επίσης τον τύπο του παθογόνου, το αναγνωριστικό της ταξινομικής ταυτότητας NCBI (εάν υπάρχει). Στην συνέχεια από τον χρήστη ζητείται να εισάγει το R0 του παθογόνου παράγοντα και την περίοδο μολυσματικότητας της νόσου, δηλαδή τι επίδραση θα έχει το παθογόνο σε πλήρες ευπαθή πληθυσμό καθώς και πόσες μέρες το παθογόνο θα μολύνει τον πληθυσμό, όπως φαίνεται και παρακάτω (έχουν εισαχθεί παντού τυχαίες τιμές για τους σκοπούς αυτού του παραδείγματος).



Εικόνα 23. Δημιουργία προσαρμοσμένου παθογόνου.

Μετά την εισαγωγή των χαρακτηριστικών του παθογόνου, το πρόγραμμα δημιουργεί το προσαρμοσμένο παθογόνο και προβάλλει στον χρήστη μια οθόνη με τα χαρακτηριστικά που εισήγαγε, όπως φαίνεται και παρακάτω:

```
Παθογόνο που επιλέχτηκε:  
Όνομα: Παράδειγμα  
Τύπος: Παράδειγμα  
Αναγνωριστικό NCBI: 123  
Αιτιώδης παράγοντας της "Παράδειγμα"  
Μέση μολυσματική περίοδος: 70.00 ημέρες  
-----
```

Εικόνα 24. Οθόνη εμφάνισης χαρακτηριστικών του παθογόνου που δημιουργήθηκε.

Σε αυτό το σημείο, το πρόγραμμα προχωράει στην δημιουργία προσαρμοσμένου πληθυσμού όπως και προηγουμένως και ζητάει από τον χρήστη να εισάγει τον συνολικό αριθμό ατόμων στον πληθυσμό, τον αριθμό ευπαθών ατόμων, τον αριθμό των αρχικά μολυσμένων ατόμων, τον αριθμό των ανοσοποιημένων ατόμων, καθώς και το μήκος προσομοίωσης σε μέρες, όπως φαίνεται και παρακάτω:

```
Μέση μολυσματική περίοδος: 70.00 ημέρες  
-----  
Δημιουργία δοκιμαστικού πλυθισμού.  
  
Εισάγετε τον συνολικό αριθμό ατόμων στον πλυθισμό: 10000  
Εισάγετε τον αριθμό των ευπαθών ατόμων: 1000  
Εισάγετε τον αριθμό των αρχικά μολυσμένων ατόμων: 100  
Εισάγετε τον αριθμό των ανοσοποιημένων ατόμων: 3  
  
Ο δοκιμαστικός πλυθισμός αποτελείται από 1000 ευπαθείς, 100 μολυσμένους,  
και 3 ανοσοποιημένους.  
  
Εισάγετε το μήκος προσομείωσης σε μέρες: 60
```

Εικόνα 25. Δημιουργία προσαρμοσμένου πληθυσμού.

Σε αυτό το σημείο, κατά την εισαγωγή και του μήκους προσομοίωσης σε ημέρες, ο αλγόριθμος ξεκινάει την προσομοίωση και δείχνει αναλυτικά κάθε μέρα της προσομοίωσης πως επηρεάστηκε ο δοκιμαστικός πληθυσμός:

```

Ημέρα 0: 1000 ευπαθείς, 100 μολυσμένοι, 3 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 1: 998.7 ευπαθείς, 99.9 μολυσμένοι, 8901.4 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 2: 997.4 ευπαθείς, 99.8 μολυσμένοι, 8902.8 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 3: 996.1 ευπαθείς, 99.7 μολυσμένοι, 8904.2 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 4: 994.8 ευπαθείς, 99.6 μολυσμένοι, 8905.6 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 5: 993.5 ευπαθείς, 99.5 μολυσμένοι, 8907.0 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 6: 992.2 ευπαθείς, 99.3 μολυσμένοι, 8908.5 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 7: 990.9 ευπαθείς, 99.1 μολυσμένοι, 8910.0 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 8: 989.6 ευπαθείς, 98.9 μολυσμένοι, 8911.5 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 9: 988.3 ευπαθείς, 98.7 μολυσμένοι, 8913.0 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 10: 987.0 ευπαθείς, 98.5 μολυσμένοι, 8914.5 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 11: 985.8 ευπαθείς, 98.3 μολυσμένοι, 8915.9 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 12: 984.6 ευπαθείς, 98.1 μολυσμένοι, 8917.3 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 13: 983.4 ευπαθείς, 97.9 μολυσμένοι, 8918.7 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 14: 982.2 ευπαθείς, 97.7 μολυσμένοι, 8920.1 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 15: 981.0 ευπαθείς, 97.5 μολυσμένοι, 8921.5 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 16: 979.8 ευπαθείς, 97.3 μολυσμένοι, 8922.9 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 17: 978.6 ευπαθείς, 97.1 μολυσμένοι, 8924.3 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 18: 977.4 ευπαθείς, 96.9 μολυσμένοι, 8925.7 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 19: 976.2 ευπαθείς, 96.7 μολυσμένοι, 8927.1 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 20: 975.0 ευπαθείς, 96.5 μολυσμένοι, 8928.5 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 21: 973.8 ευπαθείς, 96.3 μολυσμένοι, 8929.9 ανοσοποιημένοι, 10000 συνολικά.

```

Εικόνα 26. Προσομοίωση επιδημίας.

Από την παραπάνω εικόνα παρατηρούμε ότι εφόσον η περίοδος μολυσματικότητας του παθογόνου που δημιουργήσαμε είναι 70 μέρες και το μήκος προσομοίωσης σε μέρες είναι 60, τότε μέχρι και την τελευταία μέρα της προσομοίωσης, δεν θα έχουν ανοσοποιηθεί όλοι οι μολυσμένοι στον πληθυσμό.

```

Ημέρα 51: 939.3 ευπαθείς, 90.3 μολυσμένοι, 8970.4 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 52: 938.2 ευπαθείς, 90.1 μολυσμένοι, 8971.7 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 53: 937.1 ευπαθείς, 89.9 μολυσμένοι, 8973.0 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 54: 936.0 ευπαθείς, 89.7 μολυσμένοι, 8974.3 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 55: 934.9 ευπαθείς, 89.5 μολυσμένοι, 8975.6 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 56: 933.8 ευπαθείς, 89.3 μολυσμένοι, 8976.9 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 57: 932.7 ευπαθείς, 89.1 μολυσμένοι, 8978.2 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 58: 931.6 ευπαθείς, 88.9 μολυσμένοι, 8979.5 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 59: 930.5 ευπαθείς, 88.7 μολυσμένοι, 8980.8 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 60: 929.4 ευπαθείς, 88.5 μολυσμένοι, 8982.1 ανοσοποιημένοι, 10000 συνολικά.

```

Εικόνα 27. Προσομοίωση επιδημίας.

Τέλος, πριν τον τερματισμό του αλγορίθμου, το πρόγραμμα παρουσιάζει στον χρήστη τα στατιστικά στοιχεία (παθογόνο R0 σε πλήρως ευαίσθητο πληθυσμό, περίοδο μολυσματικότητας, όριο ευαισθησίας, πόσα άτομα χρειάστηκαν για την πρόληψη της επιδημίας, threshold) της προσομοίωσης.

```

Στατιστικά στοιχεία:

Παθογόνο R0 σε πλήρως ευαίσθητο πληθυσμό: 9.0
Παθογόνος περίοδος μολυσματικότητας: 70.00 ημέρες
Όριο ευαισθησίας: 1110.0 άτομα
Ανοσολογικά άτομα που χρειάστηκαν για την πρόληψη της επιδημίας: 8890.0 άτομα
Όριο κατωφλίου: 0.111

-----

Process finished with exit code 0

```

Εικόνα 28. Αποτελέσματα αλγορίθμου και τερματισμός προσομοίωσης.

Σε αυτό το σημείο ο αλγόριθμος έχει τερματίσει επιτυχώς.

4.2 ΠΕΡΙΓΡΑΦΗ ΚΩΔΙΚΑ

Μέχρι αυτό το σημείο έχουν αναλυθεί όλοι οι σημαντικοί παράγοντες που χρειάστηκαν για την δημιουργία του προγράμματος το οποίο προσομοιώνει και μοντελοποιεί την εξάπλωση μιας επιδημίας. Σε αυτό το σημείο θα αναλύσουμε πιο περιγραφικά των κώδικα του προγράμματος για την καλύτερη κατανόηση της λειτουργίας του. Όπως προαναφέρθηκε, το πρόγραμμα υλοποιήθηκε με τη χρήση γλώσσας προγραμματισμού Python. Το πρόγραμμα λειτουργεί με δύο αρχεία .py(python) το ένα ονομάζεται **epidhmia.py** και το άλλο **pathogono.py**. Στο ένα αρχείο (epidhmia.py) υπάρχει το μενού πλοήγησης για την μοντελοποίηση της επιδημίας, το επιδημιολογικό μοντέλο SIR, καθορισμός και υπολογισμός όλων των απαραίτητων μεταβλητών (για να μπορέσει να ξεκινήσει η προσομοίωση), όπως φαίνεται και παρακάτω:

```
#!/usr/b
in/env
python

# -*- coding: UTF-8 -*-

from decimal import *
from pathogono import pathDef

# Προεπιλεγμένα παθογόνα έτοιμα για χρήση
p1 = pathDef("Ιός του Ιλαρά", "Ιός", "11234",
             "Ιλαράς", 18, 8)
p2 = pathDef("Πρωτόζωο", "Ιός", "5855", "Ελονοσία", 5,
             15)
p3 = pathDef("Βάκιλος του Γερσίν", "Βακτήριο", "632",
             "Πανούκλα", 16, 6)
p4 = pathDef("Πολιοϊός", "Ιός", "12079",
             "Πολιομυελίτιδα", 10, 12)
p5 = pathDef("Ιός του Έμπολα", "Ιός", "128947",
             "Έμπολα", 8, 18)
```

```

def popCreate(p):
    print("-----")
    print("\nΠαθογόνο που επιλέχτηκε:")
    p.printPath()
    t = 10
    print("-----")
    print("Δημιουργία δοκιμαστικού πληθυσμού.\n")
    n = int(input("Εισάγετε τον συνολικό αριθμό ατόμων
στον πληθυσμό: ")) # n είναι ο συνολικός πληθυσμός
    s = int(input("Εισάγετε τον αριθμό των ευπαθών
ατόμων: ")) # s είναι ο susceptible(ευπαθής)
πληθυσμός
    i = int(input("Εισάγετε τον αριθμό των αρχικά
μολυσμένων ατόμων: ")) # i είναι ο
infectious(μολυσμένος) πληθυσμός
    r = int(input("Εισάγετε τον αριθμό των
ανοσοποιημένων ατόμων: ")) # r είναι ο
recovered/immune(ανοσοποιημένος) πληθυσμός
    print("\nΟ δοκιμαστικός πληθυσμός αποτελείται από
{0} ευπαθείς, {1} μολυσμένους, ".format(str(s),
str(i)))
    print("και {0} ανοσοποιημένους.
\n".format(str(r)))
    t1 = int(input("Εισάγετε το μήκος προσομείωσης σε
μέρες: ")) # t είναι το time(ημέρες) που θα διαρκέσει
η προσομοίωση
    if t1 != 0: # default μήκος σε ημέρες είναι το 10
και επιλέγεται με 0
        t = t1
        b = p.getB()
        y = 1 / p.getPeriod()
    print("-----")
    # εκτύπωσε τα αποτελέσματα
    print("Ημέρα 0: {0} ευπαθείς, {1} μολυσμένοι, {2}
ανοσοποιημένοι, {3} συνολικά.".format(str(s), str(i),
str(r), str(n)))
    # εφαρμογή του SIR μοντέλου
    for day in range(1, t + 1):
        s1 = s
        i1 = i
        r1 = r
        if (s1 - ((b * s1 * i1) / n)) < 0:
            s = 0.0

```

```

else:
    s = round((s1 - ((b * s1 * i1) / n)), 1)
    i = round((i1 + (((b * s1 * i1) / n) - (y
* i1))), 1)
    r = n - s - i
    # εκτύπωσε τα αποτελέσματα
    print("Ημέρα {0}: {1} ευπαθείς, {2}
μολυσμένοι, {3} ανοσοποιημένοι, {4}
συνολικά.".format(str(day), str(s),str(i),
str(r),str(n)))
    r0 = float(p.getR0())
    ip = p.getPeriod()
    th = round((1 / r0), 3)
    si = round((th * n), 1)
    ii = n - si
    print("-----")
    print("Στατιστικά στοιχεία: \n")
    print("Παθογόνο R0 σε πλήρως ευαίσθητο πληθυσμό:
{0}.".format(str(r0)))
    print("Παθογόνος περίοδος μολυσματικότητας: {0}
ημέρες".format(str(ip)))
    print("Όριο ευαισθησίας: {0}
άτομα".format(str(si)))
    print("Ανοσολογικά άτομα που χρειάστηκαν για την
πρόληψη της επιδημίας: {0} άτομα".format(str(ii)))
    print("Όριο κατωφλίου: {0}.".format(str(th)))
    print("-----\n")

# arxiko menu epilogs
print("Καλώς ήλθατε!\n")
print("1. Προσαρμοσμένος πλυθησμός με προεπιλεγμένο
παθογόνο παράγοντα.")
print("2. Προσαρμοσμένος πλυθησμός με προσαρμοσμένο
παθογόνο παράγοντα.")
print("3. Έξοδος.\n")
swch = int(input("Παρακαλώ επιλέξτε: "))
# αν επιλέχτηκε το 1 τότε
if swch == 1:
    print("-----")
    print("Προεπιλεγμένα παθογόνα \n")

```

```

    print("Παρακαλώ επιλέξτε ένα παθογόνο της παρακάτω
λίστας:\n")
    print("1.Ιλαράς")
    print("2.Πρωτόζωο")
    print("3.Βάκιλος του Γερσίν")
    print("4.Πολιοϊός")
    print("5.Έμπολα")
    ps = int(input("Παρακαλώ επιλέξτε: "))
    if ps == 1:
        pc = p1
        popCreate(pc)
    # αλλιώς αν επιλέχτηκε το 2 τότε
    elif ps == 2:
        pc = p2
        popCreate(pc)
    elif ps == 3:
        pc = p3
        popCreate(pc)
    elif ps == 4:
        pc = p4
        popCreate(pc)
    elif ps == 5:
        pc = p5
        popCreate(pc)

elif swch == 2:
    print("-----")
    print("Προσαρμοσμένη λειτουργία παθογόνου \n")
    name = input("Εισάγετε το όνομα του παθογόνου: ")
    pType = input("Εισάγετε τον τύπο του παθογόνου: ")
    taxID = input("Εισάγετε το αναγνωριστικό
ταξινομικής ταυτότητας NCBI(αν υπάρχει): ")
    disease = input("Εισάγετε το όνομα της νόσου: ")
    R0 = input("Εισάγετε το R0 του παθογόνου
παράγοντα: ")
    per = input("Εισάγετε την περίοδο
μολυσματικότητας: ")
    pc = pathDef(name, pType, taxID, disease, R0, per)
    popCreate(pc)
    # αλλιώς αν επιλέχτηκε το 3 τότε
elif swch == 3:
    print("")

```

```

        # διαφορετικά έγινε κάποιο λάθος
    else:
        print("-----")
        print("Έγινε κάποιο λάθος, παρακαλώ δοκιμάστε
        ξανά.")

    print("*****
    *****")

```

Το άλλο αρχείο (pathogono.py), χρησιμοποιείται για να καθοριστούν τα βασικά χαρακτηριστικά του παθογόνου με το οποίο θα πραγματοποιηθεί η προσομοίωση, όπως φαίνεται και παρακάτω.

```

#!/usr/bin/env
python

# -*- coding: UTF-8 -*-

from decimal import *

class pathDef(object):
    def __init__(self, path_name, path_type, taxid,
    syndrome_name, R0, ip):
        self.path_name = path_name
        self.path_type = path_type
        self.taxid = taxid
        self.syndrome_name = syndrome_name
        self.R0 = int(R0)
        a = Decimal(ip)
        self.ip = round(a,2)

    def printPath(self):
        print("Όνομα: " + self.path_name)
        print("Τύπος: " + self.path_type)
        print("Αναγνωριστικό NCBI: " + self.taxid)
        print("Αιτιώδης παράγοντας της " + "'" +
self.syndrome_name + "'")
        print("Μέση μολυσματική περίοδος: " +
str(self.ip) + " ημέρες")

```

```
def getR0(self):  
    return self.R0  
  
def getB(self):  
    b = self.R0 * (1/self.ip)  
    return b  
  
def getPeriod(self):  
    return self.ip
```

Μέσα από την συνεργασία αυτών των δύο αρχείων γίνεται εφικτή η μοντελοποίηση και η προσομοίωση η οποία περιγράφει αναλυτικά πως εξαπλώνεται μια επιδημία σε έναν δοκιμαστικό πληθυσμό. Μέσω αυτής της περιγραφής, αλλά και μέσω της έρευνας που πραγματοποιήθηκε στα προηγούμενα κεφάλαια, μπορούμε να διεξάγουμε κάποια συμπεράσματα.

4.3 ΣΥΜΠΕΡΑΣΜΑΤΑ

Μετά από όλη την ανάλυση και τα πειράματα μέσα από την χρησιμοποίηση του προγράμματος προσομοίωσης μιας επιδημίας, συμπεραίνεται ότι η προσομοίωση στον τομέα της επιδημίας αλλά και άλλων διαφόρων ασθενειών έχει μεγάλες προοπτικές και μπορεί να φανεί εξαιρετικά χρήσιμη για την αντιμετώπιση, καταπολέμηση αλλά και πρόβλεψη της εξάπλωσης ασθενειών. Φανταστείτε η παραπάνω εφαρμογή για την οποία συγγράφεται αυτή η αναφορά, να είχε εφαρμογές στον υγειονομικό τομέα της χώρας όπως για παράδειγμα την προσομοίωση και μοντελοποίηση της εξέλιξης μιας ασθένειας, π.χ την άνοια στον άνθρωπο. Αυτό θα βοηθούσε στην αναγνώριση των σταδίων της εν λόγω ασθένειας, και την προετοιμασία των νοσοκομείων /γιατρών στην καλύτερη αντιμετώπιση τους, μιας και η προσομοίωση θα εκτιμά την εξέλιξη της ασθένειας και θα δίνει χρήσιμες πληροφορίες και αποτελέσματα στον γιατρό, εκπαιδεύοντάς τον έτσι στο να αντιμετωπίζει την άνοια. Ακόμη, η προσομοίωση και μοντελοποίηση επιδημίας, μπορεί να εφαρμοστεί σε πλαίσια εκτός του υγειονομικού τομέα, μπορεί πολύ εύκολα να εφαρμοστεί σε ένα δίκτυο υπολογιστών με έναν επιδημικό ιό που εξαπλώνεται από υπολογιστή σε υπολογιστή μέσα στο δίκτυο. Επίσης, η προσομοίωση και

μοντελοποίηση, με τη χρήση του επιδημιολογικού μοντέλου **SIR**, μπορεί εύκολα να εφαρμοστεί στις περισσότερες επιδημικές ασθένειες.

Παρ' όλες τις εφαρμογές που αναφέρθηκαν, υπάρχει αρκετός χώρος για ανάπτυξη και βελτίωση. Αρχικά, στο παραπάνω αναλυθέν πρόγραμμα, μπορεί να εφαρμοστεί τεχνητή νοημοσύνη και για παράδειγμα, ο υπολογιστής που θα προσομοιώνει μια ασθένεια, μέσα από τα συμπεράσματα της προσομοίωσης, μπορεί να παίρνει αποφάσεις όπως για παράδειγμα την χορήγηση των ανάλογων φαρμάκων για την ορθή ανοσοποίηση του ατόμου. Όλα αυτά βέβαια, πρέπει πάντα να γίνονται με την έγκριση ενός εμπειρογνώμων γιατρού ο οποίος θα ελέγχει τη κατάσταση, διότι πέρα από τα θετικά που μας δίνει η τεχνητή νοημοσύνη και τα αυτόματα συστήματα, υπάρχουν αρκετοί κίνδυνοι.

Συμπερασματικά, ο τομέας της προσομοίωσης και μοντελοποίησης βρίσκεται στην κατάλληλη χρονική περίοδο έτσι ώστε να αξιοποιηθεί στο μέγιστο και να μας προσφέρει δυνατότητες που δεν μπορούσαμε σαν ανθρωπότητα να φανταστούμε προηγουμένως. Επίσης, μέσα από την δημιουργία της πτυχιακής μου, ανέπτυξα τις γνώσεις μου πάνω στην γλώσσα προγραμματισμού **Python**, καθώς επίσης ανακάλυψα και χρησιμοποίησα διάφορα εργαλεία για την δημιουργία του συγκεκριμένου αλγορίθμου. Επιπροσθέτως, εξοικειώθηκα παραπάνω με την πλατφόρμα **github**, την οποία χρησιμοποίησα για τη συνεχή ανάπτυξη και ενημέρωση της πτυχιακής μου, προσθαφαιρώντας αρχεία, πηγές, και άλλες λεπτομέρειες.

ΕΛΛΗΝΙΚΗ ΒΙΒΛΙΟΓΡΑΦΙΑ

Εξομοίωση και *προσομοίωση*,
<http://users.sch.gr/symfo/sholio/sizitontas/o/17.exomiosi.htm>, τελευταία πρόσβαση
25/9/2019.

Καλλιγερόπουλος, Δ. & Βασιλειάδου, Σ. (2005). *Συστήματα Αυτομάτου Ελέγχου Ι*,
Αθήνα: Σύγχρονη Εκδοτική, 18.

Λεβεντέας, Δ. (2008). *Οδηγός Python Μέσω Παραδειγμάτων*, 50-67.

Μαλατέστας, Π. (2011). *ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ*, Αθήνα: ΤΖΙΟΛΑ, 20-
21.

Ράπτη, Α. (2014). *Μελέτη Εξάπλωσης Ιών Σε Δίκτυα (Διπλωματική Εργασία)*, 42-50.

Ρουμελιώτης, Μ. (2001). *Μοντελοποίηση και προσομοίωση*, Πάτρα: ΤΥΡΟΡΑΜΑ, 10-
11.

Φυτίλης, Γ. (2017). *Μαθήματα Προγραμματισμού Python. Μάθημα 6^ο-7^ο*,
<http://www.pcsteps.gr>, τελευταία πρόσβαση 27/9/2019.

ΞΕΝΟΓΛΩΣΣΗ ΒΙΒΛΙΟΓΡΑΦΙΑ

Altizer, S., Nunn, C. (2006). *Infectious diseases in primates: behavior, ecology and evolution*. Oxford Series in Ecology and Evolution. Oxford: Oxford University Press.

Bailey, Norman T. J. (1975). *The mathematical theory of infectious diseases and its applications* (2nd ed.). London: Griffin.

Bequette, W. (1998). *Process Dynamics Modeling, Analysis and Simulation*, New Jersey: Prentice Hall, 21-22.

Beutel, A., et. al. (2012). *Interacting viruses in networks: Can both survive?* In Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, pages 426–434, New York, NY, USA, 2012. ACM.

Compartmental models in epidemiology (2008).

[https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The SIR model is dynamic in three senses](https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The_SIR_model_is_dynamic_in_three_senses), τελευταία πρόσβαση 25/9/2019.

Epidemiology, https://en.wikipedia.org/wiki/Epidemiology#cite_note-Epidemiology_2009-38, τελευταία πρόσβαση 27/9/2019.

Hethcote, H. (2000). *The Mathematics of Infectious Disease*, SIAM Review, 42(4): 599–653.

Hethcote, H. W. (1989). *Three Basic Epidemiological Models*, *Applied Mathematical Ecology*, Vol. 18. Berlin: Springer.

Modeling and Simulation, https://en.wikipedia.org/wiki/Modeling_and_simulation, τελευταία πρόσβαση 25/9/2019.

Prakash B. A., et. al. (2011). *Threshold conditions for arbitrary cascade models on arbitrary networks*. In Proceedings of the 2011 IEEE 11th International Conference on Data Mining, ICDM '11, 537–546, Washington, DC, USA, 2011. IEEE Computer Society.

Python (2005).<https://el.wikipedia.org/wiki/Python>, τελευταία πρόσβαση 27/9/2019.

The MSEIR model (2008).

https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The_MSEIR_model, τελευταία πρόσβαση 27/9/2019.

The SEIR model (2008).

https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The_SEIR_model, τελευταία πρόσβαση 27/9/2019

The SEIS model (2008).

https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The_SEIS_model, τελευταία πρόσβαση 27/9/2019.

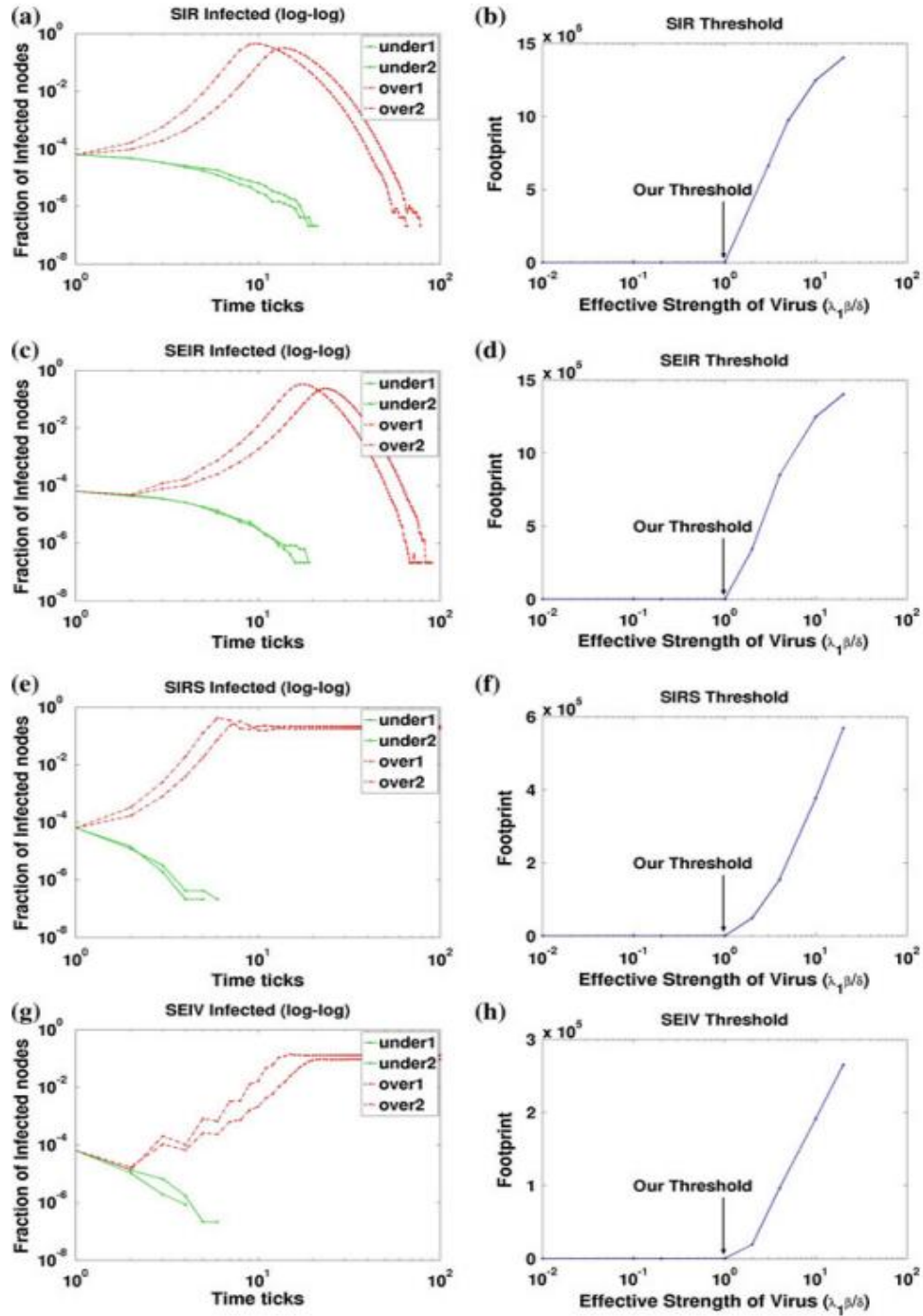
The SIR model (2008).

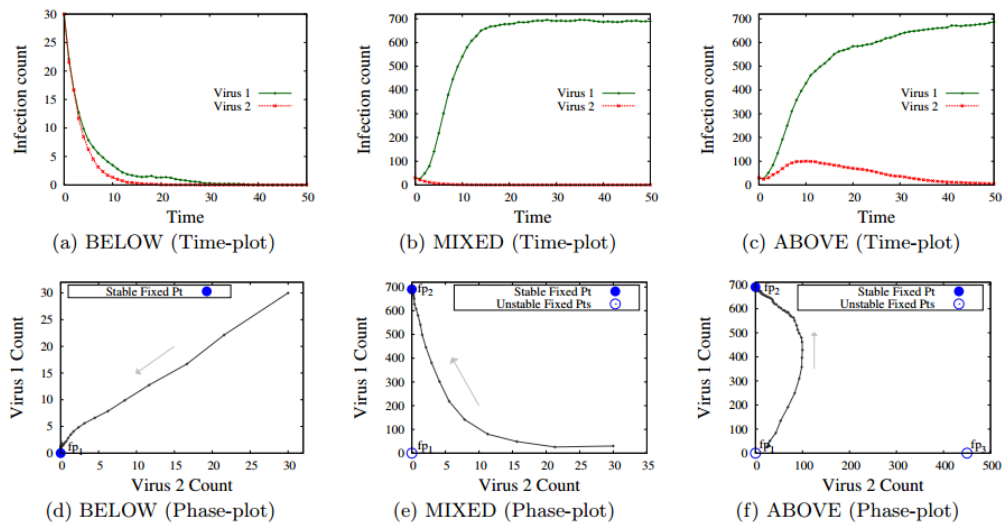
https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The_SIR_model, τελευταία πρόσβαση 27/9/2019.

The SIS model (2008).

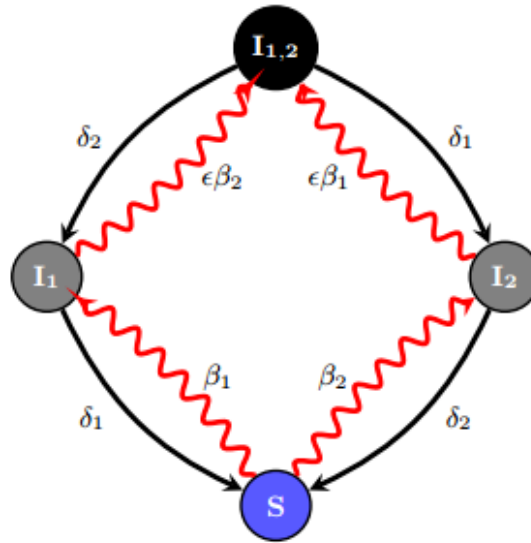
https://en.wikipedia.org/wiki/Compartmental_models_in_epidemiology#The_SIS_model, τελευταία πρόσβαση 27/9/2019.

ΠΑΡΑΡΤΗΜΑ ΕΙΚΟΝΩΝ

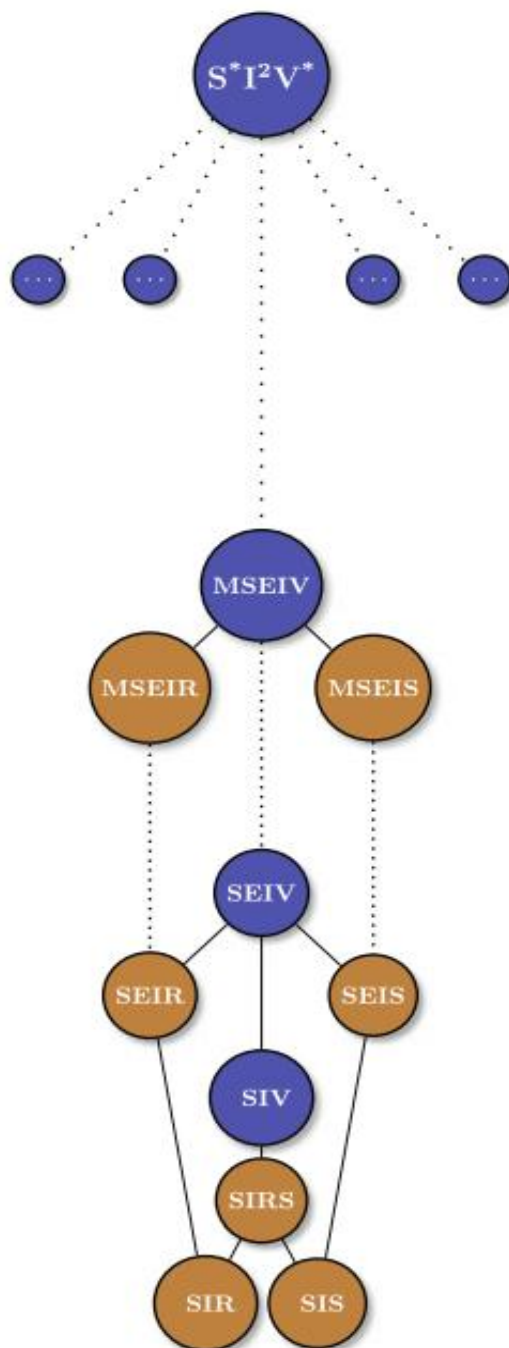
Εικόνα 1. Πειραματικά αποτελέσματα για τα επιμέρους μοντέλα του $S * I 2V *$



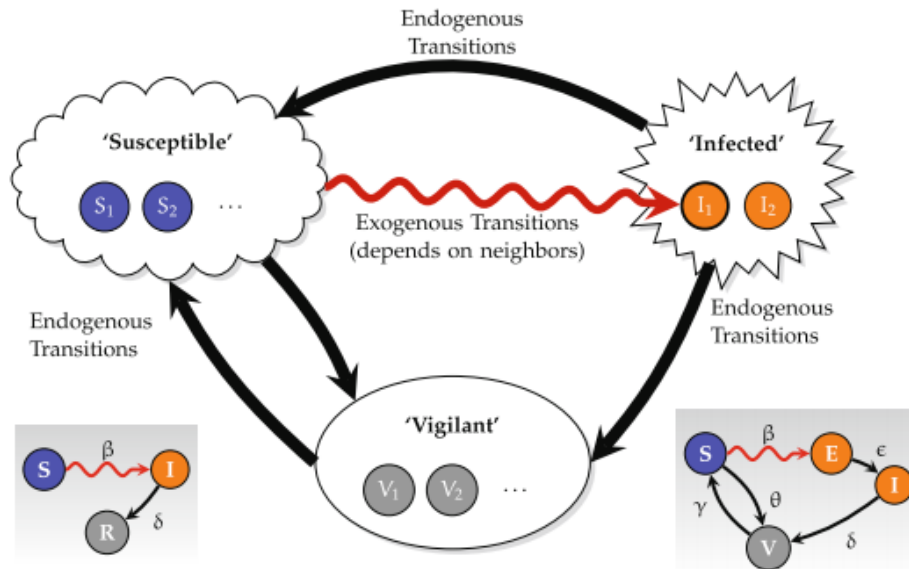
Εικόνα 2. Πειράματα στο δίκτυο AS-OREGON για τα 3 σημεία ισορροπίας (αριθμός μολυσμένων κόμβων vs χρόνος, αριθμός μολυσμένων κόμβων ιού 1 vs αριθμός μολυσμένων κόμβων ιού 2)



Εικόνα 3. Διάγραμμα καταστάσεων ενός κόμβου στο μοντέλο μερικού ανταγωνισμού S



Εικόνα 4. Γενικευμένο επιδημιολογικό μοντέλο $S^*I^2V^*$



Εικόνα 5. Διάγραμμα καταστάσεων & μεταβάσεις στο επιδημιολογικό μοντέλο SIV

Τελεστής	Περιγραφή
lambda	Εκφραση Lambda
or	Λογικό Η
and	Λογικό ΚΑΙ
not x	Λογικό ΟΧΙ
in, not in	Ελεγχoi συμμετοχής
is, is not	Ελεγχoi ταυτότητας
<, <=, >, >=, !=, ==	Συγκρίσεις
	Δυναδικό Η
^	Δυναδικό αποκλειστικό Η
&	Δυναδικό ΚΑΙ
<<, >>	Μεταθέσεις
+, -	Πρόσθεση και αφαίρεση
*, /, //, %	Πολλαπλασιασμός, Διάρθρωση, Διάρθρωση στρογγυλοποιημένη προς τα κάτω, Υπόλοιπο
+x, -x	Θετικό, Αρνητικό
~x	Δυναδικό ΟΧΙ
**	Υψωση σε δύναμη
x.attribute	Αναφορά σε χαρακτηριστικό
x[index]	Subscription
x[index1:index2]	Κομμάτισμα (Slicing)
f(arguments ...)	Κλήση συνάρτησης
(expressions, ...)	Συνένωση ή εμφάνιση πλειάδας
[expressions, ...]	Προβολή λίστας
{key:datum, ...}	Προβολή λεξικού

Εικόνα 6. Τελεστές στην Python

Τελεστής	Όνομα	Εξήγηση	Παραδείγματα
+	Συν	Προσθέτει δύο αντικείμενα.	To <code>3 + 5</code> δίνει <code>8</code> . To <code>'a' + 'b'</code> δίνει <code>'ab'</code> .
-	Μείον	Είτε δίνει έναν αρνητικό αριθμό, ή αφαιρεί έναν αριθμό από έναν άλλο.	To <code>-5.2</code> δίνει έναν αρνητικό αριθμό. To <code>50 - 24</code> δίνει <code>26</code> .
*	Επί	Δίνει το γινόμενο δύο αριθμών ή μία συμβολοσειρά (string) επαναλαμβανόμενη τόσες φορές.	To <code>2 * 3</code> δίνει <code>6</code> . To <code>'la' * 3</code> δίνει <code>'lalala'</code> .
**	Δύναμη	Επιστρέφει το x υψωμένο στη δύναμη y.	To <code>3 ** 4</code> δίνει <code>81</code> (δηλαδή <code>3 * 3 * 3 * 3</code>).
/	Διά	Διαιρεί το x με το y.	To <code>4 / 3</code> δίνει <code>1.3333333333333333</code> .
//	Διαίρεση στρογγυλοποιημένη προς τα κάτω (Floor Division)	Επιστρέφει τον κοντινότερο (προς τα κάτω) ακέραιο στο πηλίκο.	To <code>4 // 3</code> δίνει <code>1</code> .
%	Υπόλοιπο	Επιστρέφει το υπόλοιπο της διαίρεσης.	To <code>8 % 3</code> δίνει <code>2</code> . To <code>-25.5 % 2.25</code> δίνει <code>1.5</code> .
<<	Αριστερή μετάθεση	Μεταθέτει τα δυαδικά ψηφία (bits) του αριθμού προς τα αριστερά κατά το πλήθος των θέσεων που καθορίστηκε. (Κάθε αριθμός αναπαρίσταται στη μνήμη με δυαδικά ψηφία (bits, binary digits) -δηλαδή με 0 και 1).	To <code>2 << 2</code> δίνει <code>8</code> . Το 2 αναπαρίσταται ως σε bits ως <code>10</code> . Η μετάθεση προς τα αριστερά κατά 2 bits μας δίνει <code>1000</code> που παριστάνει το δεκαδικό <code>8</code> .
>>	Δεξιά μετάθεση	Μεταθέτει τα bits του αριθμού προς τα δεξιά κατά το πλήθος των θέσεων που καθορίστηκε.	To <code>11 >> 1</code> δίνει <code>5</code> . Το 11 αναπαρίσταται σε bits ως <code>1011</code> που όταν μετατεθούν δεξιά κατά 1 bit μας δίνει <code>101</code> το οποίο είναι το δεκαδικό <code>5</code> .
&	Δυαδικό ΚΑΙ	Δυαδικό ΚΑΙ των αριθμών.	To <code>5 & 3</code> δίνει <code>1</code> .
	Δυαδικό Η	Δυαδικό Η των αριθμών.	To <code>5 3</code> δίνει <code>7</code> .
^	Δυαδικό αποκλειστικό Η	Δυαδικό αποκλειστικό Η των αριθμών.	To <code>5 ^ 3</code> δίνει <code>6</code> .

Εικόνα 7. Τελεστές στην Python

~	Δυαδική αντιστροφή	Το δυαδικό αντίστροφο του x είναι $-(x+1)$.	Το <code>~5</code> δίνει <code>-6</code> .
<	Μικρότερο από	Επιστρέφει το αν το x είναι μικρότερο από το y. Όλοι οι τελεστές σύγκρισης επιστρέφουν True (Αληθής) ή False (Ψευδής). Σημειώστε ότι τα ονόματα αυτά ξεκινούν με κεφαλαίο.	Το <code>5 < 3</code> δίνει <code>False</code> και το <code>3 < 5</code> δίνει <code>True</code> . Οι συγκρίσεις μπορούν να συνδυαστούν αλυσιδωτά κατά βούληση: Το <code>3 < 5 < 7</code> δίνει <code>True</code> .
>	Μεγαλύτερο από	Επιστρέφει το αν το x είναι μεγαλύτερο από το y.	Το <code>5 > 3</code> επιστρέφει <code>True</code> . Αν και οι δύο τελεστές είναι αριθμοί, πρώτα μετατρέπονται σε έναν κοινό τύπο. Αλλιώς, επιστρέφει πάντα <code>False</code> .
<=	Μικρότερο ή ίσο	Επιστρέφει το αν το x είναι μικρότερο από ή ίσο με το y.	Το <code>x = 3; y = 6; x <= y</code> επιστρέφει <code>True</code> .
>=	Μεγαλύτερο ή ίσο	Επιστρέφει το αν το x είναι μεγαλύτερο από ή ίσο με το y.	Το <code>x = 4; y = 3; x >= 3</code> επιστρέφει <code>True</code> .
=	Ίσο	Συγκρίνει αν τα αντικείμενα είναι ίσα.	Το <code>x = 2; y = 2; x == y</code> επιστρέφει <code>True</code> . Το <code>x = 'str'; y = 'str'; x == y</code> επιστρέφει <code>False</code> . Το <code>x = 'str'; y = 'str'; x == y</code> επιστρέφει <code>True</code> .
!=	Διαφορετικό	Συγκρίνει αν τα αντικείμενα ΔΕΝ είναι ίσα.	Το <code>x = 2; y = 3; x != y</code> επιστρέφει <code>True</code> .
not	Λογικό ΟΧΙ	Αν το x είναι True, επιστρέφει False. Αν το x είναι False, επιστρέφει True.	Το <code>x = True; not x</code> επιστρέφει <code>False</code> .
and	Λογικό ΚΑΙ	Το x and y επιστρέφει False αν το x είναι False, αλλιώς επιστρέφει υπολογίζει και επιστρέφει την τιμή του y.	Το <code>x = False; y = True; x and y</code> επιστρέφει <code>False</code> αφού το x είναι <code>False</code> . Σε αυτή την περίπτωση, η Python δε θα ελέγξει την τιμή του y αφού γνωρίζει ότι η αριστερή πλευρά της έκφρασης <code>and</code> είναι <code>False</code> που υποδηλώνει ότι ολόκληρη η έκφραση θα είναι <code>False</code> ανεξάρτητα από τις άλλες τιμές. Αυτή η τεχνική αποκαλείται <i>short-circuit evaluation</i> .
or	Λογικό Ή	Αν το x είναι True, επιστρέφει True, αλλιώς υπολογίζει και επιστρέφει την τιμή του y.	Το <code>x = True; y = False; x or y</code> επιστρέφει <code>True</code> . Η <i>short-circuit evaluation</i> εφαρμόζεται και εδώ.

Εικόνα 8. Τελεστές στην Python

```
>>> 2 * (4 - 1)
6
>>> 2 * 4 - 1
7
>>> (2 + 1) ** (5 - 2)
27
>>> 2 ** 1 + 1
3
>>> 2 ** (1 + 1)
4
>>>
```

Εικόνα 9. Προτεραιότητα τελεστών

```
>>> 3 * 5 + 2
17
>>> 15 / 3 - 2
3.0
>>>
```

Εικόνα 10. Προτεραιότητα τελεστών

```
File Edit Format Run Options Window Help
# Υπολογισμός του εμβαδού ενός ορθογωνίου

length = 5 # μήκος
width = 6 # πλάτος
area = length * width # υπολογισμός εμβαδού
print('Εμβαδόν =', area) # εκτύπωση
```

Εικόνα 11. Παράδειγμα σχολίων

```
age = 21
if age >= 18:
    print("You vote")
else:
    print("You don't vote")
```

Εικόνα 12. Παράδειγμα προγράμματος

```
from itertools import count
def generate_primes(stop_at=None):
    primes = []
    for n in count(2):
        if stop_at is not None and n > stop_at:
            return
        composite = False
        for p in primes:
            if not n % p:
                composite = True
                break
            elif p**2 > n:
                break
        if not composite:
            primes.append(n)
            yield n
```

Εικόνα 13. Παράδειγμα προγράμματος

```
for i in generate_primes():
    if i > 100: break
    print(i)
```

Εικόνα 14. Παράδειγμα συνάρτησης υπολογισμού πρώτων αριθμών από 1 έως 100

```

class Employee:
    def __init__(self, id, name, position):
        self.id = id;
        self.name = name;
        self.position = position;
    def printEmployee(self):
        print(self.id, "|", self.name, "|", self.position);

e1 = Employee(1230, "Νικόλαος Παπαγεωργίου", "Βοηθός Προγραμματιστή");
e1.printEmployee();

```

Εικόνα 15. Παράδειγμα κλάσης "υπάλληλος"

```

Connected to pydev debugger (build 192.5728.105)
Καλώς ήλθατε!

1. Προσαρμοσμένος πληθυσμός με προεπιλεγμένο παθογόνο παράγοντα.
2. Προσαρμοσμένος πληθυσμός με προσαρμοσμένο παθογόνο παράγοντα.
3. Έξοδος.

Παρακαλώ επιλέξτε:

```

Εικόνα 16. Αρχικό μενού πλοήγησης του αλγορίθμου.

```

Παρακαλώ επιλέξτε: 1
-----
Προεπιλεγμένα παθογόνα

Παρακαλώ επιλέξτε ένα παθογόνο της παρακάτω λίστας:

1. Ιλαράς
2. Πρωτόζωο
3. Βάκτιλος του Γερσίν
4. Πολιοϊός
5. Έμπολα
Παρακαλώ επιλέξτε:

```

Εικόνα 17. Προσαρμοσμένος πληθυσμός με προεπιλεγμένο παθογόνο παράγοντα.

```

Παθογόνο που επιλέχτηκε:
Όνομα: Ιός του Ιλαρά
Τύπος: Ιός
Αναγνωριστικό NCBI: 11234
Αιτιώδης παράγοντας της "Ιλαράς"
Μέση μολυσματική περίοδος: 8.00 ημέρες
-----

```

Εικόνα 18. Οθόνη προβολής πληροφοριών παθογόνου.

```

-----
Δημιουργία δοκιμαστικού πλυθισμού.

Εισάγετε τον συνολικό αριθμό ατόμων στον πλυθισμό: 10000
Εισάγετε τον αριθμό των ευπαθών ατόμων: 1000
Εισάγετε τον αριθμό των αρχικά μολυσμένων ατόμων: 100
Εισάγετε τον αριθμό των ανοσοποιημένων ατόμων: 2

Ο δοκιμαστικός πλυθισμός αποτελείται από 1000 ευπαθείς, 100 μολυσμένους,
και 2 ανοσοποιημένους.

```

Εικόνα 19. Οθόνη προβολής συνολικής αναφοράς εισαχθέντων τιμών.

```

Εισάγετε το μήκος προσομοίωσης σε μέρες: 70
-----
Ημέρα 0: 100 ευπαθείς, 100 μολυσμένοι, 2 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 1: 97.8 ευπαθείς, 89.8 μολυσμένοι, 9812.4 ανοσοποιημένοι, 10000 συνολικά.
Ημέρα 2: 95.8 ευπαθείς, 80.6 μολυσμένοι, 9823.6 ανοσοποιημένοι, 10000 συνολικά.

```

Εικόνα 20. Εισαγωγή μήκους προσομοίωσης σε ημέρες.

```

Στατιστικά στοιχεία:

Παθογόνο R0 σε πλήρως ευαίσθητο πληθυσμό: 5.0
Παθογόνος περίοδος μολυσματικότητας: 15.00 ημέρες
Όριο ευαισθησίας: 2000.0 άτομα
Ανοσολογικά άτομα που χρειάστηκαν για την πρόληψη της επιδημίας: 8000.0 άτομα
Όριο κατωφλίου: 0.2
-----

Process finished with exit code 0

```

Εικόνα 21. Αποτελέσματα αλγορίθμου και τερματισμός της προσομοίωσης.

```

Καλώς ήλθατε!

1. Προσαρμοσμένος πληθυσμός με προεπιλεγμένο παθογόνο παράγοντα.
2. Προσαρμοσμένος πληθυσμός με προσαρμοσμένο παθογόνο παράγοντα.
3. Έξοδος.

Παρακαλώ επιλέξτε: 2
-----
Προσαρμοσμένη λειτουργία παθογόνου

Εισάγετε το όνομα του παθογόνου: |

```

Εικόνα 22. Προσαρμοσμένος πληθυσμός με προσαρμοσμένο παθογόνο παράγοντα.

```

Εισάγετε το όνομα του παθογόνου: Παράδειγμα
Εισάγετε τον τύπο του παθογόνου: Παράδειγμα
Εισάγετε το αναγνωριστικό ταξινομικής ταυτότητας NCBI(αν υπάρχει): 123
Εισάγετε το όνομα της νόσου: Παράδειγμα
Εισάγετε το R0 του παθογόνου παράγοντα: 9
Εισάγετε την περίοδο μολυσματικότητας: 70
-----

```

Εικόνα 23. Δημιουργία προσαρμοσμένου παθογόνου.

```
Παθογόνο που επιλέχτηκε:  
Όνομα: Παράδειγμα  
Τύπος: Παράδειγμα  
Αναγνωριστικό NCBI: 123  
Αιτιώδης παράγοντας της "Παράδειγμα"  
Μέση μολυσματική περίοδος: 70.00 ημέρες  
-----
```

Εικόνα 24. Οθόνη εμφάνισης χαρακτηριστικών του παθογόνου που δημιουργήθηκε.

```
Μέση μολυσματική περίοδος: 70.00 ημέρες  
-----  
Δημιουργία δοκιμαστικού πληθυσμού.  
  
Εισάγετε τον συνολικό αριθμό ατόμων στον πληθυσμό: 10000  
Εισάγετε τον αριθμό των ευπαθών ατόμων: 1000  
Εισάγετε τον αριθμό των αρχικά μολυσμένων ατόμων: 100  
Εισάγετε τον αριθμό των ανοσοποιημένων ατόμων: 3  
  
Ο δοκιμαστικός πληθυσμός αποτελείται από 1000 ευπαθείς, 100 μολυσμένους,  
και 3 ανοσοποιημένους.  
  
Εισάγετε το μήκος προσομείωσης σε μέρες: 60
```

Εικόνα 25. Δημιουργία προσαρμοσμένου πληθυσμού.

Ημέρα 0: 1000 ευπαθείς, 100 μολυσμένοι, 3 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 1: 998.7 ευπαθείς, 99.9 μολυσμένοι, 8901.4 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 2: 997.4 ευπαθείς, 99.8 μολυσμένοι, 8902.8 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 3: 996.1 ευπαθείς, 99.7 μολυσμένοι, 8904.2 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 4: 994.8 ευπαθείς, 99.6 μολυσμένοι, 8905.6 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 5: 993.5 ευπαθείς, 99.5 μολυσμένοι, 8907.0 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 6: 992.2 ευπαθείς, 99.3 μολυσμένοι, 8908.5 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 7: 990.9 ευπαθείς, 99.1 μολυσμένοι, 8910.0 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 8: 989.6 ευπαθείς, 98.9 μολυσμένοι, 8911.5 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 9: 988.3 ευπαθείς, 98.7 μολυσμένοι, 8913.0 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 10: 987.0 ευπαθείς, 98.5 μολυσμένοι, 8914.5 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 11: 985.8 ευπαθείς, 98.3 μολυσμένοι, 8915.9 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 12: 984.6 ευπαθείς, 98.1 μολυσμένοι, 8917.3 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 13: 983.4 ευπαθείς, 97.9 μολυσμένοι, 8918.7 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 14: 982.2 ευπαθείς, 97.7 μολυσμένοι, 8920.1 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 15: 981.0 ευπαθείς, 97.5 μολυσμένοι, 8921.5 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 16: 979.8 ευπαθείς, 97.3 μολυσμένοι, 8922.9 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 17: 978.6 ευπαθείς, 97.1 μολυσμένοι, 8924.3 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 18: 977.4 ευπαθείς, 96.9 μολυσμένοι, 8925.7 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 19: 976.2 ευπαθείς, 96.7 μολυσμένοι, 8927.1 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 20: 975.0 ευπαθείς, 96.5 μολυσμένοι, 8928.5 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 21: 973.8 ευπαθείς, 96.3 μολυσμένοι, 8929.9 ανοσοποιημένοι, 10000 συνολικά.

Εικόνα 26. Προσομοίωση επιδημίας.

Ημέρα 51: 939.3 ευπαθείς, 90.3 μολυσμένοι, 8970.4 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 52: 938.2 ευπαθείς, 90.1 μολυσμένοι, 8971.7 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 53: 937.1 ευπαθείς, 89.9 μολυσμένοι, 8973.0 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 54: 936.0 ευπαθείς, 89.7 μολυσμένοι, 8974.3 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 55: 934.9 ευπαθείς, 89.5 μολυσμένοι, 8975.6 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 56: 933.8 ευπαθείς, 89.3 μολυσμένοι, 8976.9 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 57: 932.7 ευπαθείς, 89.1 μολυσμένοι, 8978.2 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 58: 931.6 ευπαθείς, 88.9 μολυσμένοι, 8979.5 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 59: 930.5 ευπαθείς, 88.7 μολυσμένοι, 8980.8 ανοσοποιημένοι, 10000 συνολικά.
 Ημέρα 60: 929.4 ευπαθείς, 88.5 μολυσμένοι, 8982.1 ανοσοποιημένοι, 10000 συνολικά.

Εικόνα 27. Προσομοίωση επιδημίας.

```
Στατιστικά στοιχεία:  
  
Παθογόνο R0 σε πλήρως ευαίσθητο πληθυσμό: 9.0  
Παθογόνος περίοδος μολυσματικότητας: 70.00 ημέρες  
Όριο ευαισθησίας: 1110.0 άτομα  
Ανοσολογικά άτομα που χρειάστηκαν για την πρόληψη της επιδημίας: 8890.0 άτομα  
Όριο κατωφλίου: 0.111  
-----  
  
Process finished with exit code 0
```

Εικόνα 28. Αποτελέσματα αλγορίθμου και τερματισμός προσομοίωσης.